Redundancy Support

- Feature Summary and Revision History, on page 1
- High Availability Support, on page 2
- Geographic Redundancy Support, on page 5

Feature Summary and Revision History

Summary Data

Table 1: Summary Data

Applicable Product(s) or Functional Area	cnSGW-C
Applicable Platform(s)	SMI
Feature Default Setting	Enabled - Configuration Required
Related Documentation	Not Applicable

Revision History

Table 2: Revision History

Revision Details	Release
First introduced.	2021.02.0

High Availability Support

Feature Description

The cnSGW-C is built on the Kubernetes cluster strategy so that it inherits the high availability aspects of K8 cluster deployments. The cnSGW-C uses the construct that includes the components, such as pods and services.

Each pod has at least 2 instances to ensure high availability against:

- Pod instance restart or failure
- Pod lost due to node restart or failure

For details on the pods and services, see the Pods and Services Reference chapter in this guide.

High Availability of UDP Proxy

The cnSGW-C supports High Availability (HA) of UDP proxy. The HA model of UDP proxy is based on the keepalived virtual IP concepts.

For more information on UDP proxy redundancy, see the High Availability for the UDP Proxy section in the Pods and Services Reference chapter.

Architecture

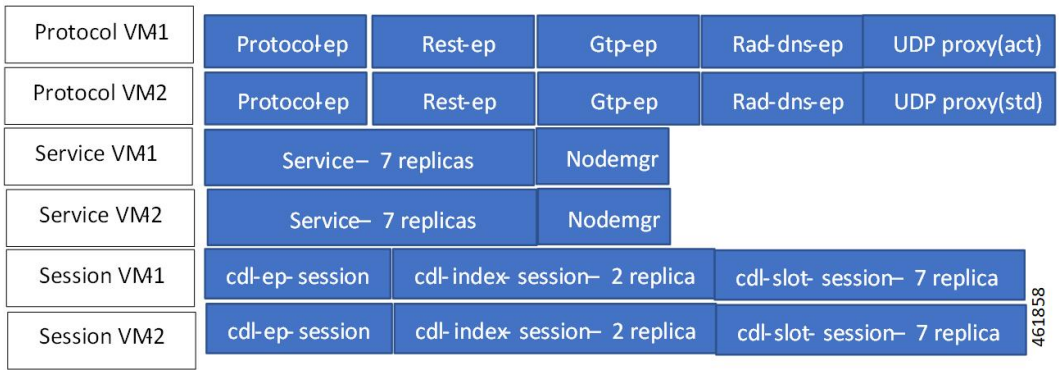
This section describes the recommended layout of cnSGW-C pods and VMs.

cnSGW-C Pod and VM Deployment Layout

This section describes the deployment of cnSGW-C pods and its microservices.

The following figure shows the deployment model of six VMs in cnSGW-C.

Figure 1: VM Deployment Model



In this model, the pods are deployed on VM pairs. Two replicas are available for each protocol pod (for example, rest-ep, protocol-ep, and gtp-ep). One instance is deployed on each protocol VM.

Similarly, service pods and session pods are distributed equally on both the service and session VMs. Such a distribution is controlled by labelling the VMs as well as implementing the K8 affinity and anti-affinity rules during pod scheduling.

This model ensures that, during VM reboot scenarios, at least 50% of the replicas of each pod type are available to handle user signaling.

Graceful pod restart allows pod to complete ongoing processing within 30 seconds. Abrupt pod restart will affect ongoing transactions without impact to PDU sessions.

How it Works

This section provides information on how the resiliency and HA can be achieved.

The cnSGW-C enables inter-pod communication during the pod failure or restart.

During graceful pod restart:

- Ongoing processing is not impacted.
- New messages are not sent to the pod through Kubernetes service.
- Messages with session affinity continue to be received by the pod.
- Existing call flow expected to complete within 30 seconds.

After pod restart:

- All Prometheus metrics of the pod are reset.
- When internal diagnostics is green, the pod status changes to Ready.
- Pod is ready to process the new messages.

When the cnSGW-C VM reboots or the VM is unavailable:

- All pods on the VM are lost.
- Pods on the other available VM continue processing, thus providing high availability.
- VIP, if present, is switched to the other available node.
- It takes about 5 minutes of the node unreachability for Kubernetes to detect the node as down.
- Pods on the node are thereafter not discoverable through Kubernetes service.

After the pod restarts, pods on the VM are scheduled one after another. This operation is similar to the pod restart.

During the VIP and VM reboot, virtual IP is associated with a single VM. UDP proxy binds to N4 VIP address for communication with UPF. UDP proxy binds to S5 VIP address for communication with cnSGW-C.

Reboot of VM with active VIP causes VIP to switch to the other protocol VM. The active UDP proxy failure causes VIP to switch to other protocol VM.

Before the Subscriber Microservices Infrastructure (SMI) handles the VIP monitoring and switchover, make sure that appropriate VIP configuration is available in the SMI deployer. Also, check if the port is set to 28000 and the host priority is equal.

Configuring Pod-level Labelling and Replicas

The node label is configured on the SMI cluster deployer. For information on the configuration commands, see the Mapping Pods with Node Labels section in the Deploying and Configuring cnSGW-C through Operations Center chapter.

Configuration Example

The following is an example of VM labelling and replica configuration.

```
k8 label protocol-layer key smi.cisco.com/node-type value smf-proto
exit
k8 label service-layer key vm-type value smf-svc
exit
k8 label cdl-layer key smi.cisco.com/node-type value smf-cdl
exit
k8 label oam-layer key smi.cisco.com/node-type value oam
exit

endpoint pfcp
 replicas 1
 nodes    2
exit
endpoint service
 replicas 1
 nodes    2
exit
endpoint protocol
 replicas 1
 nodes    2
 vip-ip 209.165.201.28
exit
endpoint sbi
 replicas 1
 nodes    2
```

Configuration Verification

To verify the configuration, use the following show command:

show running-config instance instance-id *instance_id* **endpoint**

The following is an example output of this show command.

```
show running-config instance instance-id 1 endpoint
instance instance-id 1
 endpoint nodemgr
  replicas 1
  nodes    2
 exit
 endpoint gtp
  replicas 1
  vip-ip 209.165.201.29
 exit
 endpoint pfcp
  replicas                2
  enable-cpu-optimization true
  interface n4
   heartbeat
    interval               0
    retransmission-timeout 3
    max-retransmissions    5
```

```
  exit
 exit
exit
endpoint service
 replicas 2
exit
endpoint protocol
 replicas 1
 vip-ip 209.165.201.29
 end
```

This command output displays the configurations related to multiple endpoints, such as endpoint names, pod replicas, and nodes.

Geographic Redundancy Support

Feature Description

The cnSGW-C supports Geographical Redundancy (GR) in active-active mode. The GR takes place through replication of sessions, configuration, and any other data required for seamless failover and failback of services to the remote site.

How it Works

cnSGW-C (CNF) can be deployed in a geographic redundant manner to provide service across a catastrophic failure, such as data center failure of a rack hosting a cnSGW-C cluster.

Each CNF instance service registers with NRF and S11/S5 for DNS entry for MME/SGW. Local HA redundancy allows instance to achieve rack level redundancy in addition to K8 cluster level failures within same data center or handle locally within same K8 cluster if failed containers are per Type-2 < *n*.

where, *n* is a value. For less than 50% of container failures, HA should handle the failures. For more than 50% of container failures, GR switchover is triggered.

Overview

In active-active mode,

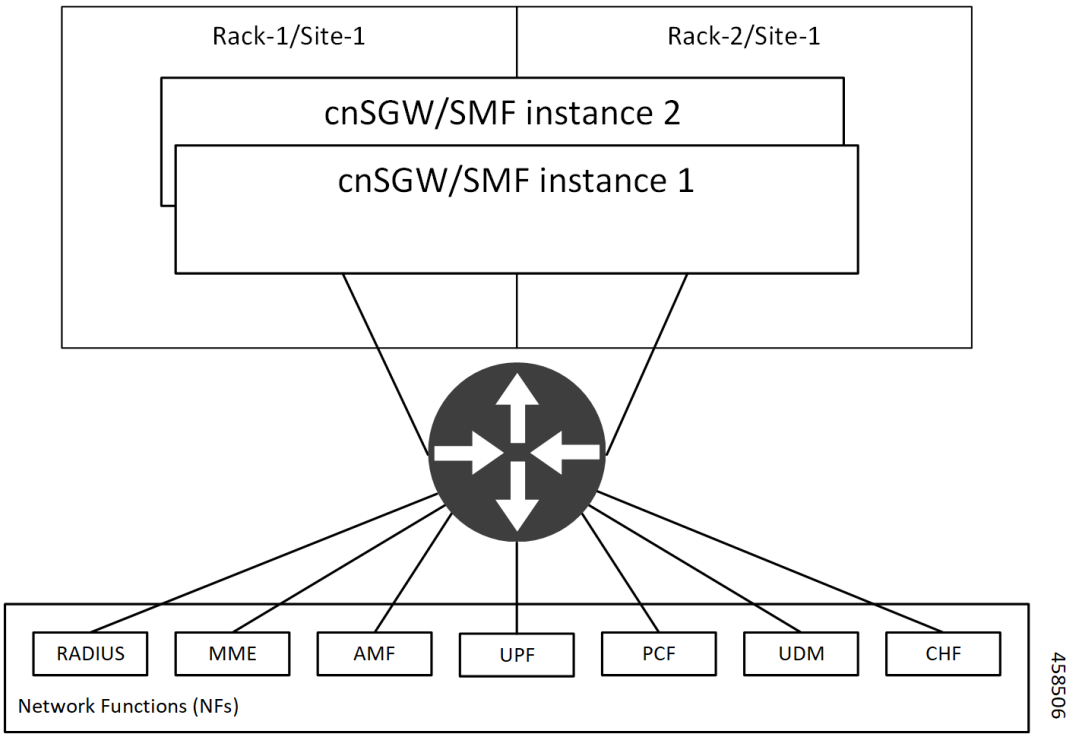
- GR deployment is transparent to the adjacent NFs.
- GR deployment contains two instances of CCG function, each instance manifest itself with a set of interface IPs.
- Each instance support sets of sessions and continue to use the same IP for session consistency.
- At a specific time period, one CCG instance can be primary only on one site and standby on the other site.
- The set of interface IPs that are associated with the CCG instance, dynamically route to the primary site of the instance.

cnSGW-C supports primary/standby redundancy in which data is replicated from the Primary to standby instance. The primary instance provides services in normal operation. If the primary instance fails, the standby instance becomes the primary and takes over operation. To achieve a geographically distributed system, two

primary/standby pairs can be set up where each site is actively processing traffic and standby is acting as backup for the remote site.

In an Active/Active GR deployment, consider there are two racks: Rack-1/Site-1 and Rack-2/Site-2 located either locally or remotely from each other in separate geographic site. All the NFs are trying to reach instance-1 and instance-2.

Figure 2: Active-Active GR Deployment



For NFs, both the instances are active. But in actual, instance-1 and instance-2 are divided across racks.

Rack-1/Site-1 has instance-1 and instance-2. In a pre-trigger scenario, instance-1 is local and acts as Primary and instance-2 is in Standby mode.

Rack-2/Site-2 also has instance-1 and instance-2. In a pre-trigger scenario, instance-2 is local and acts as Primary and instance-1 is in Standby mode.

In case, if Rack-1/Site-1 goes down, the traffic moves to Rack-2/Site-2. On Rack-2/Site-2 both the instances, instance-1 and instance-2 acts as Primary.

GR Triggers

Geographic Redundancy (GR) supports the following GR triggers:

- CLI-based switchover: Manual CLI commands are used to switch the roles and trigger GR failover.
- BFD link failover detection: When both the BFD links between the connected rack and the leafs are down, GR failover is triggered.
- Local Site POD failure detection: When threshold percentage of POD replica-sets failing is greater than the configured threshold value, GR failover is triggered.

- Remote Site POD failure detection: When the remote POD monitoring detects failure breaching threshold percentage, the POD becomes self-primary for that instance.
- Remote Site Role monitoring: When the remote role monitoring detects that the rack is in Standy_error state, it becomes self-primary.
- Multi-Compute Failure: When two or more servers are powered down, it triggers GR failover.

Site Roles

Note
- Cachepod/ETCD and CDL Replication happens during all the roles mentioned in the following section.
- If GR links are down or periodic heartbeat fails, GR triggers are suspended.

- PRIMARY: Site is ready and actively taking traffic for the given instance.
- STANDBY: Site is standby, ready to take traffic but not taking traffic for the given instance.
- STANDBY_ERROR: Site is in problem, not active and not ready to take traffic for the given instance.
- FAILOVER_INIT: Site has started to failover and not in condition to take traffic. Buffer time is 2 sec for application to complete their activity.
- FAILOVER_COMPLETE: Site has completed the failover and attempted to inform the peer site about the failover for given instance. Buffer time is 2 seconds.
- FAILBACK_STARTED: Manual failover is triggered with delay from remote site for the given instance.

For fresh installation, site boots up with:

- Role PRIMARY for local instance (each site has local instance-id configured to identify local instance). It is recommended not to configure the pods for monitoring during fresh installation. Once the setup is ready, you can configure the pods for monitoring.
- Role STANDBY for other instances

For upgrades, site boots up with:

- STANDBY_ERROR role for all the instances as moving the traffic post upgrade needs manual intervention.
- ETCD stores instance roles.

Note Rolling upgrade or in-service upgrade isn't supported.

General Guidelines

Before configuring Geographic Redundancy deployment, here are some general guidelines:

- Both GR sites should be on the same software version.
- Both GR sites should be configured with same configuration.
- Loopback port of Instance 1 and Instance 2 should be different. Else, REST-EP POD wouldn't come up due to K8 IP/Port conflict.
- Respective interface on both GR sites should be on the same VLAN. For example, N4 VLAN of Instance1 and Instance2 should be on the same VLAN. Else, there's a route conflict on Kernel while enforcing BGP policies.
- Consult your Cisco Technical Representative to perform the following procedures to make sure proper roles are assigned.

  For more information, see Software Upgrade on GR Pairs, on page 34.
- Post GR, perform the failback manually after ensuring the site is healthy. Autonomous failback isn't supported.

  For more information, see Recovery Procedure, on page 55.
- Use non-bonded interface in BGP speaker PODs for BGP peering.
- BGP peering per Proto node is supported with only two BGP routers/leafs. Considering two Proto nodes, there can be maximum of four BGP neighborships.
- Use bonded interfaces for Service traffic.
- Geo pod uses two VIPs:
  - Internal-VIP for Inter-POD communication (within the rack)
  - External-VIP for Inter-Rack Geo pod communication. Configure only on Proto Nodes on L2 Subnet. This is used to communicate across the racks. This node has external connectivity to other Rack
- Geo Internal IP to be reachable to all nodes within the rack.
- Geo External IP:
- CDL/Kafka VIPs: Configure on CDL Labeled Nodes on L2 Subnet.
- Enable LI tapping on both sites.
- MDF server should be reachable from both sites.

Instance Awareness

Instance awareness configuration in cnSGW-C helps to distinguish local site instance and remote site instance.

Configuring GR Instance

This configuration is needed to provide a geo-redundancy configuration for multiple sites. With instance ID, endpoint configurations should be configured for each Geo-Redundancy site.

Sample Configuration 1

The following is a sample configuration for endpoint VIP configuration under one instance:

```
config
  instance instance-id gr_instanceId
    endpoint endpoint_name
        vip-ip vip_ip_address
  exit
exit
```

Example:

```
config
instance instance-id 1
 endpoint sbi
  vip-ip 209.165.201.21
 exit
exit
```

Sample Configuration 2

The following is a sample configuration to provide information on system-id, cluster-id and slice-name under an instance:

```
config
  instances instance instance_id
   system-id system_id
   cluster-id cluster_id
   slice-name cdl_slice_name
  exit
exit
```

Example:

```
config
 instances instance 1
  system-id sgw
  cluster-id sgw
  slice-name 1
 exit
exit
```

Note It is recommended to have the same values for *system-id*, *cluster-id* in the instance, and *app-name*, *cluster-name* in deployment.

Configuring Endpoint Instance Awareness

Only two instances can be configured on each local and remote site, and corresponding endpoints can be instantiated.

A local instance-id is the identity of the local site irrespective of the site is in GR aware or not.

Local Instance ID Configuration

The local instance is configured using the local-instance command.

```
local-instance instance 1
```

Endpoint configuration must be under instance specified by each unique instance ID.

Endpoint Configuration Example

Following are a few configuration examples.

Note In the following example, *instance-id* "**1**" is a local instance-id, and endpoints configured under it belong to the local site.

Optionally, remote site *instance-id* "**2**" can be configured for endpoints belonging to the geo-site.

```
instance instance-id 1
 endpoint li
  replicas 1
  nodes    2
  vip-ip 209.165.201.6
  vip-ip 209.165.201.13
 exit
 endpoint gtp
  replicas 1
  nodes    2
  retransmission timeout 5 max-retry 4
  vip-ip 209.165.201.6
  vip-ip 209.165.201.4
  interface s5
   echo interval          60
   echo retransmission-timeout 5
   echo max-retransmissions 4
  exit
  interface s2b
   echo interval 60
   echo retransmission-timeout 5
   echo max-retransmissions 4
  exit
 exit
exit
instance instance-id 2
 endpoint li
  replicas 1
  nodes    2
  vip-ip 209.165.201.6
  vip-ip 209.165.201.13
 exit
exit
endpoint gtp
  replicas 1
  nodes    2
  retransmission timeout 5 max-retry 4
  vip-ip 209.165.201.6
  vip-ip 209.165.201.5
  interface s5
   echo interval 60
   echo retransmission-timeout 5
   echo max-retransmissions 4
  exit
  interface s2b
   echo interval 60
   echo retransmission-timeout 5
   echo max-retransmissions 4
  exit
 exit
exit
```

Configuring Profile cnSGW-C Instance Awareness

Add instance for PGW FQDN corresponding to local and remote instances.

Example

Following is a configuration example.

Note In the following example, *instance-id* "**1**" is a local instance-id, and the cnSGW-C profile configured under it belongs to the local site.

Optionally, remote site *instance-id* "**2**" can be configured for FQDN belonging to the geo-site.

```
profile sgw sgw1
locality LOC1
instances 1 fqdn cisco.com.apn.epc.mnc456.mcc123
instances 2 fqdn cisco.com.apn.epc.mnc567.mcc123
```

Configuring cnSGW-C Endpoint

Endpoint configuration is required only for cnSGW-C.

Example

The following is a configuration example.

Note In the following example, *instance-id* "**1**" is a local instance-id, and endpoints configured under it belong to the local site.

Optionally, remote site *instance-id* "**2**" can be configured for endpoints belonging to the geo-site.

```
instance instance-id 1
 endpoint nodemgr
  replicas 1
  nodes    1
 exit
 endpoint gtp
  replicas 1
  vip-ip 209.165.201.10
  interface s5e
   vip-ip 209.165.201.29
  exit
  interface s11
   vip-ip 209.165.201.29
  exit
 exit
 endpoint pfcp
  replicas 1
  interface sxa
   heartbeat
    interval               0
    retransmission-timeout 5
    max-retransmissions    3
   exit
  exit
```

```
exit
 endpoint service
  replicas 1
 exit
 endpoint protocol
  replicas 1
  vip-ip 209.165.201.29
  interface sxa
   vip-ip 209.165.201.29
  exit
 exit
 endpoint sgw-service
  replicas 1
 exit
exit
instance instance-id 2
 endpoint nodemgr
  replicas 1
  nodes    1
 exit
 endpoint gtp
  replicas 1
  vip-ip 209.165.202.150
  interface s5e
   vip-ip 209.165.201.27
  exit
  interface s11
   vip-ip 209.165.201.27
  exit
 exit
 endpoint pfcp
  replicas 1
  interface sxa
   heartbeat
    interval               0
    retransmission-timeout 5
    max-retransmissions    3
   exit
  exit
exit
 endpoint service
  replicas 1
 exit
 endpoint protocol
  replicas 1
  vip-ip 209.165.201.27
  interface sxa
   vip-ip 209.165.201.27
  exit
 exit
 endpoint sgw-service
  replicas 1
 exit
exit
```

Dynamic Routing

Border Gateway Protocol (BGP) allows you to create loop-free inter-domain routing between autonomous systems (AS). An AS is a set of routers under a single technical administration. The routers can use an Exterior Gateway Protocol to route packets outside the AS. The Dynamic Routing by Using BGP feature enables you to configure the next-hop attribute of a BGP router with alternate local addresses to service IP addresses with

priority and routes. The App-Infra BGP speaker pods enable dynamic routing of traffic by using BGP to advertise pod routes to the service VIP.

This feature supports the following functionality:

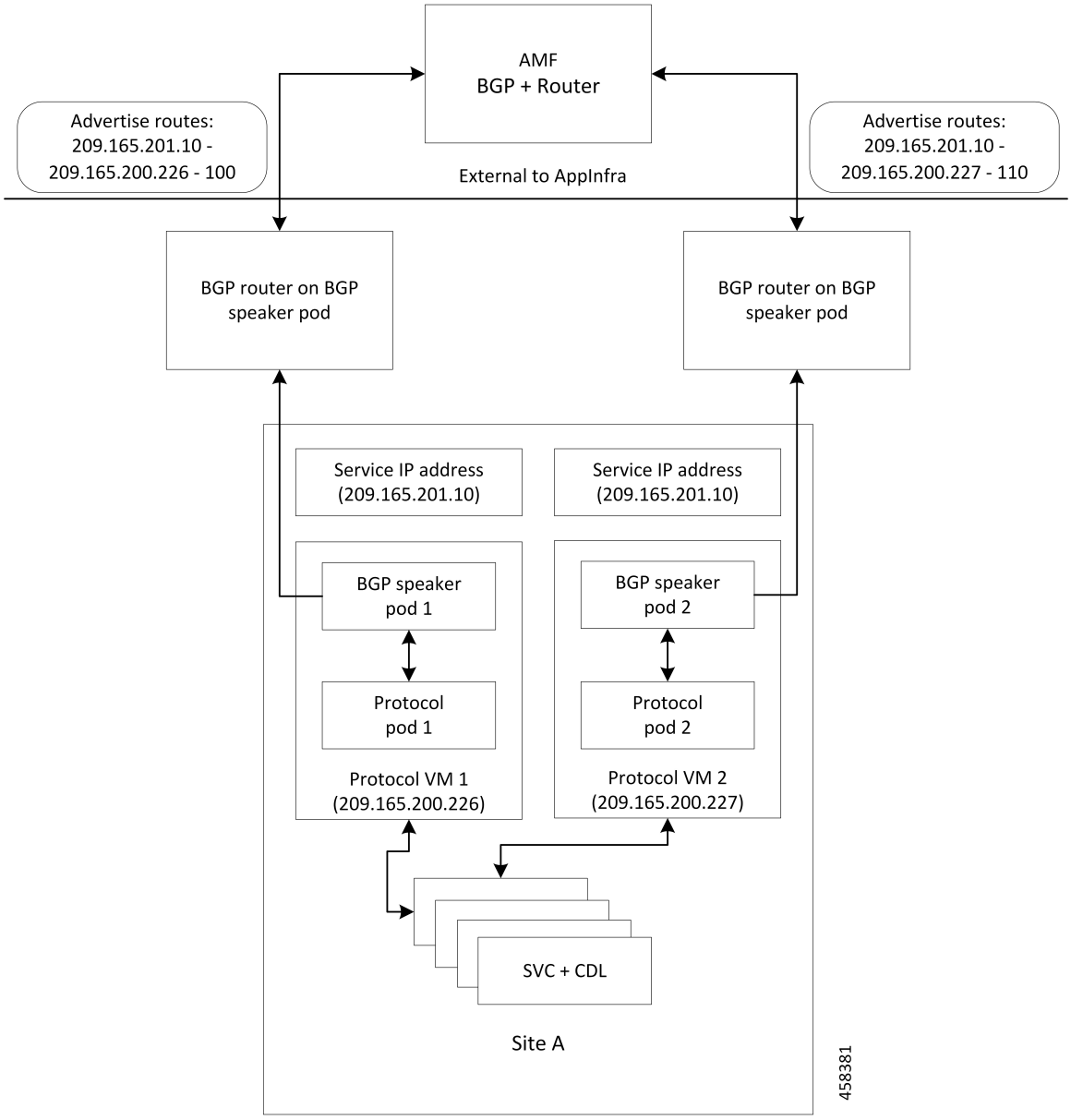
- Dynamic routing by using BGP to advertise service IP addresses for the incoming traffic.
- Learn route for outgoing traffic.
- Handling a BGP pod failover.
- Handling a protocol pod failover.
- Statistics and KPIs for the BGP speakers.
- Log messages for debugging the BGP speakers.
- Enable or disable the BGP speaker pods.
- New CLI commands to configure BGP.

Incoming Traffic

BGP uses TCP as the transport protocol, on port 179. Two BGP routers form a TCP connection between one another. These routers are peer routers. The peer routers exchange messages to open and confirm the connection parameters.

The BGP speaker publishes routing information of the protocol pod for incoming traffic in the active standby mode. Use the following image as an example to understand the dynamic routing functionality. There are two protocol pods, pod1 and pod2. Pod1 is active and pod2 is in the standby mode. The service IP address, 209.165.200.225 is configured on both the nodes, 209.165.200.226 and 209.165.200.227. pod1 is running on host 209.165.200.226 and pod2 on host 209.165.200.227. The host IP address exposes the pod services. BGP speaker publishes the route 209.165.200.225 through 209.165.200.226 and 209.165.200.227. It also publishes the preference values, 110 and 100 to determine the priority of pods.

Figure 3: Dynamic Routing for Incoming Traffic in the Active-standby Topology



For high availability, each cluster has two BGP speaker pods with Active-standby topology. Kernel route modification is done at host network level where the protocol pod runs.

MED Value

The Local Preference is used only for IGP neighbours, whereas the MED Attribute is used only for EGP neighbours. A lower MED value is the preferred choice for BGP.

Table 3: MED Value

Bonding Interface Active	VIP Present	MED Value	Local Preference
Yes	Yes	1210	2220
Yes	No	1220	2210
No	Yes	1215	2215
No	No	1225	2205

Bootstrap of BGP Speaker Pods

The following sequence of steps set up the BGP speaker pods:

1. The BGP speaker pods use TCP as the transport protocol, on port 179. These pods use the AS number configured in the Ops Center CLI.
2. Register the Topology manager.
3. Select the Leader pod. The Active speaker pod is the default choice.
4. Establish connection to all the BGP peers provided by the Ops Center CLI.
5. Publish all existing routes from ETCD.
6. Configure import policies for routing by using CLI configuration.
7. Start gRPC stream server on both the speaker pods.
8. Similar to the cache pod, two BGP speaker pods must run on each Namespace.

For more information on Dynamic Routing, see the *Dynamic Routing by Using BGP* chapter in the *UCC Serving Gateway Control Plane Function - Configuration and Administration Guide*.

Configuring Dynamic Routing by Using BGP

This section describes how to configure the Dynamic Routing by Using BGP feature.

Configuring AS and BGP Router IP Address

To configure the AS and IP address for the BGP router, use the following commands:

```
config
  router bgp local_as_number
  exit
exit
```

NOTES:

- **router bgp** *local_as_number*—Specify the identification number for the AS for the BGP router.

  In a GR deployment, you need to configure two Autonomous Systems (AS).

  - One AS for leaf and spine.
  - Second AS for both racks: Rack-1/Site-1 and Rack-2/Site-2

Configuring BGP Service Listening IP Address

To configure the BGP service listening IP address, use the following commands:

```
config
   router bgp local_as_number
      interface interface_name
   exit
exit
```

NOTES:

- **router bgp** *local_as_number*—Specify the identification number for the AS for the BGP router.
- **interface** *interface_name*—Specify the name of the interface.

Configuring BGP Neighbors

To configure the BGP neighbors, use the following commands:

```
config
   router bgp local_as_number
      interface interface_name
      neighbor neighbor_ip_address remote-as as_number
   exit
exit
```

NOTES:

- **router bgp** *local_as_number*—Specify the identification number for the AS for the BGP router.
- **interface** *interface_name*—Specify the name of the interface.
- **neighbor** *neighbor_ip_address*—Specify the IP address of the neighbor BGP router.
- **remote-as** *as_number*—Specify the identification number for the AS.

Configuring Bonding Interface

To configure the bonding interface related to the interfaces, use the following commands:

```
config
   router bgp local_as_number
      interface interface_name
      bondingInterface interface_name
   exit
exit
```

NOTES:

- **router bgp** *local_as_number*—Specify the identification number for the AS for the BGP router.
- **interface** *interface_name*—Specify the name of the interface.
- **bondingInterface** *interface_name*—Specify the related bonding interface for an interface. If the bonding interface is active, then the BGP gives a higher preference to the interface-service by providing a lower MED value.

Configuring Learn Default Route

If the user configures specific routes on their system and they need to support all routes, then they must set the **learnDefaultRoute** as **true**.

Note This configuration is optional.

To configure the Learn Default Route, use the following commands:

```
config
   router bgp local_as_number
      learnDefaultRoute true/false
   exit
exit
```

NOTES:

- **router bgp** *local_as_number*—Specify the identification number for the AS for the BGP router.
- **learnDefaultRoute** *true/false*—Specify the option to enable or disable the **learnDefaultRoute** parameter. When set to true, BGP learns default route and adds it in the kernel space. By default, it is false.

Configuring BGP Port

To configure the Port number for a BGP service, use the following commands:

```
config
   router bgp local_as_number
      loopbackPort port_number
   exit
exit
```

NOTES:

- **router bgp** *local_as_number*—Specify the identification number for the AS for the BGP router.
- **loopbackPort** *port_number*—Specify the port number for the BGP service. The default value is 179.

Policy Addition

The BGP speaker pods learns many route information from its neighbors. However, only a few of them are used for supporting the outgoing traffic. This is required for egress traffic handling only, when cnSGW-C is sending information outside to AMF/PCF. Routes are filtered by configuring import policies on the BGP speakers and is used to send learned routes to the protocol pods.

A sample CLI code for policy addition and the corresponding descriptions for the parameters are shown below.

```
$bgp policy <policy_Name> ip-prefix 209.165.200.225 subnet 16 masklength-range 21..24
as-path-set "^65100"
```

Table 4: Import Policies Parameters

Element	Description	Example	Optional
as-path-set	AS path value	"^65100"	Yes

Element	Description	Example	Optional
ip-prefix	Prefix value	“209.165.200.225/16”	Yes
masklength-range	Range of length	“21..24”	Yes
interface	Interface to set as source IP (default is VM IP)	eth0	Yes
gateWay	Change gateway of incoming route	209.165.201.30	Yes
modifySourceIp	Modify source ip of incoming route Default value is False.	true	Yes
isStaticRoute	Flag to add static IP address into kernel route Default value is False.	true	Yes

Configuring BGP Speaker

This configuration controls the number of BGP speaker pods in deployment. BGP speaker advertises service IP information for incoming traffic from both the sites.

Note

- Use non-bonded interface in BGP speaker pods for BGP peering.
- BGP peering per Proto node is supported with only two BGP routers/leafs. Considering two Proto nodes, there can be maximum of four BGP neighborships.

```
instance instance-id  instance_id endpoint bgpspeaker interface { bgp | bfd
 } internal base-port start base_port_number
```

```
config
instance instance-id instance_id
 endpoint bgpspeaker
   replicas replica_id
   nodes node_id
   interface bgp
     internal base-port start base_port_number
   exit
   interface bfd
     internal base-port start base_port_number
  exit
exit
```

NOTES:

- **instance instance-id** *instance_id*—Specify the GR instance ID.
- *base_port_number*—Specify the port range only if logical NF is configured. This range depends on your deployment.

Example

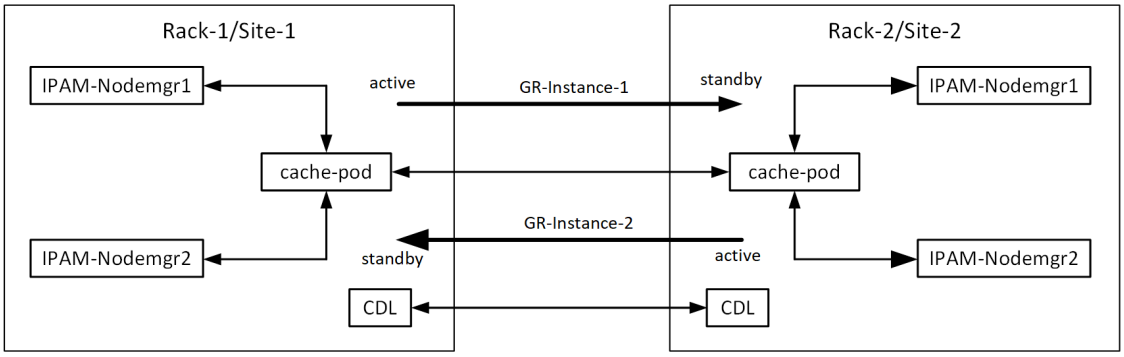
The following is a configuration example:

```
instance instance-id 1
endpoint bgpspeaker
   replicas 1
   nodes    2
   interface bgp
     internal base-port start {24000}
   exit
   interface bfd
     internal base-port start {25000}
 exit
```

IPAM

This section describes IP Address Management (IPAM) at the rack/site level.

Figure 4: IPAM



During UPF registration, active IPAM instance reserves four address-ranges per UPF per DNN.

- Range-1: Active cluster, nodemgr-1
- Range-2: Active cluster, nodemgr-2
- Range-3: Standby cluster, nodemgr-1
- Range-4: Standby cluster, nodemgr-2

During normal operation, Rack-1/Site-1 handles UPF-register/release, address-allocate/release for subscribers coming up in GR-instance-1.

If Rack-2/Site-2 goes down, Rack-1/Site-1 gets role-change trigger for GR-Instance-2.

- IPAM in Rack-1/Site-1, restores the content of GR-Instance-2 from local-cache-pod (which was already synced)
- IPAM in Rack-1/Site-1 handles UPF-Register/Release and address-allocate/release for subscribers coming up with GR-Instance-2 using the restored content in addition to handling GR-Instance-1.

Each IPAM pool is associated to a GR-Instance, with the following:

- Pool name is unique across all the instances.
- Address-ranges are unique within VRF and across all the instances.

The same pool configuration must be configured in both the active and standby cnSGW-C clusters of a particular instance.

During address-allocation, active instance assign free-IP from reserved address-range for the UPF.

Incase new address-ranges is not available, change ownership of standby's address-range to current active instance and continue assigning address-ranges from it.

Configuring IPAM

The following section provides IPAM configuraton examples.

cnSGW-C-1 Example

The following is a configuration example for cnSGW-C-1:

```
ipam
 instance 1
  address-pool pool-1
  vrf-name ISP
   tags
   dnn dnn-1
  exit
  ipv4
    address-range 209.165.201.1 209.165.201.31
 exit
instance 2
 address-pool pool-2
  vrf-name ISP
  tags
   dnn dnn-2
  exit
  ipv4
    address-range 209.165.202.129 209.165.202.159
 exit
exit
```

cnSGW-C-2 Example

The following is a configuration example for cnSGW-C-2:

```
ipam
 instance 1
 address-pool pool-1
  vrf-name ISP
  tags
   dnn dnn-1
  exit
  ipv4
    address-range 209.165.201.1 209.165.201.31
 exit
 instance 2
 address-pool pool-2
  vrf-name ISP
   tags
   dnn dnn-2
  exit
  ipv4
    address-range 209.165.202.129 209.165.202.159
 exit
exit
```

Geo Replication

Geo replication is used for inter-rack/site communication and for POD/VIP/BFD monitoring within the rack. In Geographic Redundancy,

- Two instances of Geo pods are running for each rack/site.
- Two Geo pods functions in Active-Standby mode.
- Each Geo pod instance is spawned on different Proto node/VM.
- Geo pod running on the Proto node/VM having VIP is Active Geo pod.
- In the event of Active Geo pod restart, VIPs are switched to other Proto node/VM and Standby Geo pod running on the other Proto node/VM becomes active.
- Geo pod uses host networking mode (similar to UDP-Proxy).
- Geo pod uses two VIPs:
  - Internal-VIP for Inter-POD communication (within the rack)
  - External-VIP for Inter-Rack Geo pod communication. Configure only on Proto Nodes on L2 Subnet. This is used to communicate across the racks. This node has external connectivity to other Rack.
- Logical-NF-InstanceID should be configured same for both cnSGW-Cs in GR-Pair
- For KeepAliveD monitoring,
  - Geo pod uses base port as: 15000+(Logical-NF-InstanceID * 32) + 4

    Geo pod base port should be different than BGP speaker pod port.
    - Default port (without logical cnSGW-C) would be: 15004
    - For Logical cnSGW-C configured with logical-nf-instance-id as 1, the port would be: 15036
  - UDP-Proxy pod uses base port as : 28000+Logical-NF-InstanceID
    - Default port (without logical cnSGW-C) would be: 28000
    - For Logical cnSGW-C configured with logical-nf-instance-id as 1, the port would be: 28001
  - BGPSpeaker-pod uses default base port as: 20000+(Logical-NF-InstanceID * 32)+4
    - Default port (without logical cnSGW-C) would be: 20004
    - For Logical cnSGW-C configured with logical-nf-instance-id as 1, the port would be: 20036

Note Only ETCD and cachepod data is replicated to standby rack.

Configuring ETCD/Cachepod Replication

Endpoints must be configured under an instance. Two Geo-Redundancy pods are needed on each GR site. You should also configure VIP for internal and external Geo interface for ETCD/CachePod replication.

```
instance instance-id  instance_id endpoint geo interface { geo-internal |
geo-external } vip-ip { vip_ip_address } vip-port { vip_port_number }
```

```
config
instance instance-id instance_id
 endpoint geo
  replicas replica_id
  nodes node_id
  internal base-port start base_port_number
  interface geo-internal
   vip-ip vip_ip_address vip-port vip_port_number
  exit
  interface geo-external
   vip-ip vip_ip_address vip-port vip_port_number
  exit
exit
exit
```

NOTES:

- **instance instance-id** *instance_id*—Specify GR instance ID. One instance ID for local site and other for remote site.
- **vip-ip** *vip_ip_address*—Specify VIP IP address for Internal/External Geo interface.
- **vip-port** *vip_port_number*—Specify VIP port number.
- **internal base-port start** *base_port_number*—Specify port range only if logical NF is configured.

Example

The following is a configuration example:

```
instance instance-id 1
endpoint geo
  replicas 1
  nodes 2
  internal base-port start 25000
  interface geo-internal
   vip-ip 209.165.201.8 vip-port 7001
  exit
  interface geo-external
   vip-ip 209.165.201.8 vip-port 7002
  exit
exit
```

Geo Monitoring

This section describes Geo monitoring.

POD Monitoring

To conifgure POD monitoring and failover thresholds in the GR setup, use the following configuration. The GR pod monitors the configured POD name.

```
config
 geomonitor
```

```
 podmonitor pods pod_name
  retryCount value
  retryInterval interval_value
  retryFailOverInterval failover_interval
  failedReplicaPercent percent_value
 exit
exit
```

NOTES:

- **pods** *pod_name*—Specify the name of the pod to be monitored. For example, Cache-pod, res-ep, and so on
- **retryCount** *value*—Specify the retry counter value to retry if pod fails to ping after which pod is marked as down. Must be an integer in the range of 1-10.
- **retryInterval** *interval_value*—Specify the retry interval in milliseconds if pod successfully pings. Must be an integer in the range of 200-10000.
- **retryFailOverInterval** *failover_interval*—Specify the retry interval in milliseconds if pod fails to ping. Must be an integer in the range of 200-10000.
- **failedReplicaPercent** *percent_value*—Specify the percent value of failed replica after which GR failover is triggered. Must be an integer in the range of 10-100.

Configuration Example

The following is an example configuration.

```
geomonitor podmonitor pods cache-pod
 retryCount 3
 retryInterval 5
 retryFailOverInterval 1
 failedReplicaPercent 40
exit
```

Remote Cluster Monitoring

Remote cluster monitoring auto corrects roles (it becomes self primary if the remote site is in STANDBY_ERROR state) for uninterrupted traffic flow of traffic. However, this auto role correction is done for specific roles only.

```
config
 geomonitor
  remoteclustermonitor
   retryCount value
   retryInterval interval_value
  exit
```

NOTES:

- **retryCount** *value*—Specify the retry count before making current site PRIMARY. Must be an integer in the range of 1-10. Default value is 3.
- **retryInterval** *interval_value*—Specify the retry interval in milliseconds after which remote site status is fetched. Must be an integer in the range of 200-50000. Default value is 3000.

Configuration Example

The following is an example configuration

```
geomonitor remoteclustermonitor
retryCount 3
retryInterval 3000
```

Traffic Monitoring

The following command is used to monitor the traffic.

```
config
 geomonitor
  trafficMonitor
   thresholdCount value
   thresholdInterval interval_value
  exit
 exit
```

NOTES:

- **thresholdCount** *value*—Specify the number of calls received for standby instance. Must be an integer in the range of 0-10000. Default value is 0.

  Both UDP-proxy and REST-EP must be considered for the counter value.

- **thresholdInterval** *interval_value*—Specify the maximum duration to hit the threshold count value in ms. Must be an integer in the range of 100-10000. Default value is 3000.

Configuration Example

The following is an example configuration

```
geomonitor trafficmonitor
thresholdCount 3
thresholdInterval 3000
```

BFD Monitoring

Bidirectional Forwarding Detection (BFD) protocol is used for Faster Network Failure Detection along with BGP. Whenever connectivity between BGP peering fails with cluster (NF), failover is triggered to minimize traffic failure impact.

```
config
  router bgp as
    bfd interval interval min_rx min_rx multiplier multiplier
    loopbackPort loopbackPort loopbackBFDPort loopbackBFDPort
  interface interface_id (BGP on non-bonded interface <-- loopbackEth)
    bondingInterface bondingInterface (leaf6-nic)
    bondingInterface bondingInterface (leaf6-nic)
    neighbor neighbor_ip_address remote-as remote_as fail-over fail_over_type
  exit
  interface interface_id (BGP on non-bonded interface <-- loopbackEth)
    bondingInterface bondingInterface (leaf7-nic)
    bondingInterface bondingInterface (leaf7-nic)
```

```
    neighbor bondingInterface remote-as remote_as fail-over fail_over_type
  exit
  policy-name policy_name
   as-path-set as_path_set
   gateWay gateWay_address
   interface interface_id_source
   ip-prefix ip_prefix_value
   isStaticRoute false | true
   mask-range mask_range
   modifySourceIp false | true
  exit
exit
```

NOTES:

- **bgp** *as*—Specify the Autonomous System (AS) path set.
- **bfd**—Specify BFD configuration.
  - **interval** *interval* —Specify BFD interval in milliseconds.
  - **min_rx** *min_rx*—Specify BFD minimum RX in milliseconds.
  - **multiplier** *multiplier*—Specify BFD interval multiplier.
- **interface** *interface_id*—Specify BGP local interface.
  - **bondingInterface** *bondingInterface*—Specify linked bonding interface.
  - **neighbor** *neighbor_ip_address*—Specify IP address of neighbor.
    - **fail-over** *fail_over_type*—Specify failover type.
    - **remote-as** *remote_as*—Specify Autonomous System (AS) number of BGP neighbor.
- **learnDefaultRoute**—Learn default route and add it in kernel space
- **loopbackBFDPort** *loopbackBFDPort*—Specify BFD local port.
- **loopbackPort** *loopbackPort*—Specify BGP local port.
- **policy-name** *policy_name*—Specify policy name.
  - **as-path-set** *as_path_set*—Specify Autonomous System (AS) path set.
  - **gateWay** *gateWay_address*—Specify gateway address.
  - **interface** *interface_id_source*—Specify interface to set as source IP.
  - **ip-prefix** *ip_prefix_value*—Specify IP prefix value.
  - **isStaticRoute** *false* | *true*—Specify whether to add static route in kernel space. Default value is false.
  - **mask-range** *mask_range*—Specify mask range.
  - **modifySourceIp** *false* | *true*—Modify source IP of the incoming route. Default value is false.

true: This option is used for non-UDP related VIPs. Source IP of the given interface is used as Source IP while sending out packets from cnSGW-C.

false: This option is used for all UDP related VIPs. VIP is used as Source IP while sending out packets from cnSGW-C.

Example

Following are configuration examples:

```
router bgp 65000
  bfd interval 250000 min_rx 250000 multiplier 3
  loopbackPort 179 loopbackBFDPort 3784
interface ens160 (BGP on non-bonded interface <-- loopbackEth)
  bondingInterface enp216s0f0 (leaf6-nic)
  bondingInterface enp216s0f1 (leaf6-nic)
  neighbor leaf6-ip remote-as 60000 fail-over bfd
exit
interface ens192 (BGP on non-bonded interface <-- loopbackEth)
  bondingInterface enp94s0f1 (leaf7-nic)
  bondingInterface enp94s0f0 (leaf7-nic)
  neighbor leaf7-ip remote-as 60000 fail-over bfd
exit
policy-name allow-all ip-prefix 209.165.201.30/0 mask-range 0...32
exit
```

BGP router configuration with BFD

```
show running-config router
router bgp 65142
 learnDefaultRoute false
 bfd interval 250000 min_rx 250000 multiplier 3
 interface enp94s0f0.3921
  bondingInterface enp216s0f0
  bondingInterface enp94s0f0
  neighbor 209.165.201.24 remote-as 65141 fail-over bfd
 exit
 interface enp94s0f1.3922
  bondingInterface enp216s0f1
  bondingInterface enp94s0f1
  neighbor 209.165.202.24 remote-as 65141 fail-over bfd
```

Show BFD status of neighbor

```
show bfd-neigbor
status-details

----- bgpspeaker-pod-1----

Peer             Status

209.165.202.142  STATE_DOWN
----- bgpspeaker-pod-2----

Peer             Status

209.165.202.142  STATE_UP
```

```
policy-name allow-n11 ip-prefix 209.165.200.225/54 mask-range 25..32 interface bd1.n11.2271
 modifySourceIp true isStaticRoute true gateWay 209.165.201.14
```

In the above example, *modifySourceIp* is set to true.

- AMF subnet: 209.165.200.225/54

N11 Svc Bonded Physical Interface: bd1.n11.2271 (IP address - 209.165.201.23)

N11 Svc Bonded VxLAN Anycast GW: 209.165.201.14

N11 VIP Address: 209.165.201.7

- cnSGW-C Outbound Packet (will have source IP as 209.165.201.23)

  Inbound Packet to cnSGW-C (will have destination IP as 209.165.201.7)

```
policy-name allow-n4-1 ip-prefix 209.165.201.17/41 mask-range 24..32 interface bd2.n4.2274
 gateWay 209.165.201.17
```

In the above example, *modifySourceIp* is set to false (default).

- UPF N4 Interface IP: 209.165.201.17/41

  N4 Svc Bonded Physical Interface: bd2.n4.2274 (IP address - 209.165.201.23)

  N4 Svc Bonded VxLAN Anycast GW: 209.165.201.17

  N4 VIP Address: 209.165.201.14

- cnSGW-C Outbound Packet (will have source IP as 209.165.201.14)

  Inbound Packet to cnSGW-C (will have destination IP as 209.165.201.14)

CDL GR Deployment

By default, CDL is deployed with two replicas for db-ep, 1 slot map (2 replicas per map), and 1 index map (2 replicas per map).

Note It is recommended to configure the CDL container in YANG.

Prerequisites for CDL GR

Before deploying the CDL GR, user must configure the following:

- CDL Session Database and define the base configuration.
- Kafka for CDL.
- Zookeeper for CDL.

CDL Instance Awareness and Replication

In CDL, along with existing GR related parameters, GR instance awareness must be enabled using a feature flag on all sites. Also, the mapping of system-id to slice names should also be provided for this feature to work on all sites.

The CDL is also equipped with Geo Replication (GR) failover notifications, which can notify the timer expiry of session data and bulk notifications to the currently active site. The CDL uses Border Gateway Protocol (BGP) through App-Infra for the GR failover notifications.

The CDL subscribes to the key value on both the GR sites. The App-Infra sends notifications to the CDL when there is any change in these key values. A key value indicates the state of the CDL System ID or the

GR instance. The GR instance is mapped to the CDL slices using the CDL system ID or the GR instance ID in the key.

The system ID is mandatory on both the sites. The GR instance ID in the NF configuration must match the CDL system ID.

CDL has instance-specific data slices. It also allows users to configure instance-specific slice information at the time of bringing up.

- CDL notifies the data on expiry or upon bulk notification request from the active slices.
- CDL determines the active instance based on the notification from app-infra memory-cache.
- CDL slice is a partition within a CDL instance to store a different kind of data. In this case, NF stores a different instance of data.

Note CDL slice name should match with the slice-name configured in GR.

Configuring CDL Instance Awareness

The following command is used to configure CDL instance awareness.

```
config
cdl
 datastore datastore_session_name
   features
    instance-aware-notification
      enable [ true | false ]
      system-id system_id
       slice-names slice_names
   end
```

NOTES:

- **datastore** *datastore_session_name*—Specify the datastore name.
- **enable [true | false]**—Enables the GR instance state check for slices.
- **system-id** *system_id*—Mapping of system ID to slice name.
- **slice-names** *slice_names*—Specify the list of slice names associated with the system ID. CDL slice name should match with the slice-name configured in GR.

Example

The following is a configuration example:

```
cdl datastore session
 features instance-aware-notification enable true
 features instance-aware-notification system-id 1
  slice-names [ sgw1 smf1 ]
 exit
 features instance-aware-notification system-id 2
  slice-names [ sgw2 smf2 ]
 end
```

Configuring CDL Replication

This section describes CDL replication configuration.

1. Configure Site-1 CDL HA system without any Geo-HA-related configuration parameters.
   a. Set the System ID as 1 in the configuration.
   b. Set the slot map/replica and index map/replica and Kafka replica as per requirements.

   The following is a sample configuration:

```
cdl system-id 1
cdl node-type session
cdl datastore session
endpoint replica replica_id
  slot map 4
  slot replica 2
  index map 1
  index replica 2
cdl kafka replica 2
```

1. Configure external IPs on Rack-1/Site-1 for Rack-2/Site-2 to Rack-1/Site-1 communication.
   a. Enable geo-replication on Rack-1/Site-1 and configure the remote Rack as 2 for Rack-1/Site-1.

      cdl enable-geo-replication *true*
   b. Configure the external IP for CDL endpoint to be accessed by Rack-2/Site-2.

      cdl datastore session endpoint external-ip *site-1_external_ip*
   c. Configure the external IP and port for all Kafka replicas.

      So, if two replicas (default) are configured for Kafka, user need to provide two different <*ip*>+<*port*> pairs.

      cdl kafka external-ip *site-1_external_ip port1* **cdl kafka external-ip** *site-1_external_ip port2*

2. Add remote site (Site-1) information on Rack-2/Site-2.
   - Remote site cdl-ep configuration on Rack-2/Site-2:

     cdl remote-site 1 db-endpoint host *site-1_cdl_ep_ip*

     cdl remote-site 1 db-endpoint port *site-1_cdl_ep_port*

     (Port Example: 8882)
   - Remote site Kafka configuration on Rack-2/Site-2:

     cdl remote-site *1* kafka-server *site-1_kafka1_ip site-1_kafka1_port*

     cdl remote-site *1* kafka-server *site-1_kafka2_ip site-1_kafka2_port*
   - Direct the session datastore configuration to remote Rack-2/Site-2 configuration:

     cdl datastore session geo-remote-site *1*
   - (Optional) Configure the SSL certificates to establish a secure connection with remote site on Rack-1/Site-1. All the certificates are in multi-line raw text format. If the certificates are not valid, the server continues with non-secure connection.

cdl ssl-config certs *site-2_external_ip* **ssl-key** *<ssl_key>*

cdl ssl-config certs *site-2_external_ip* **ssl-crt** *<ssl_crt>*

3. Commit GR configuration on Rack-2/Site-2:
   - Commit the configuration and let the pods be deployed on Rack-2/Site-2.
   - Verify all pods are in running state.
   - Once both sites are deployed, verify that the mirror maker pods on both sites are running and in ready state.

Examples

HA:

```
cdl node-type db-ims

cdl datastore session
 endpoint replica 2
 index map    1
 index write-factor 1
 slot replica 2
 slot map     4
 slot write-factor 1
exit

k8 label cdl-layer key smi.cisco.com/node-type value smf-ims-session
```

Rack-1/Site-1:

```
cdl system-id          1
cdl node-type          session
cdl enable-geo-replication true
cdl zookeeper replica 1

cdl remote-site 2
 db-endpoint host 209.165.201.21 >> Rack-2 external CDL IP
 db-endpoint port 8882
 kafka-server 209.165.201.21 10092 >> Rack-2 external CDL IP
 exit
exit

cdl label-config session
 endpoint key smi.cisco.com/node-type1
 endpoint value smf-cdl
 slot map 1
  key   smi.cisco.com/node-type1
  value smf-cdl
 exit
 index map 1
  key   smi.cisco.com/node-type1
  value smf-cdl
 exit
exit
cdl logging default-log-level debug

cdl datastore session
 label-config    session
 geo-remote-site [ 2 ]
 slice-names     [ 1 2 ]
```

```
 endpoint cpu-request 100
 endpoint replica 2
 endpoint external-ip 209.165.201.25 >> Rack-1 external CDL IP
 endpoint external-port 8882
 index cpu-request 100
 index replica 2
 index map   1
 slot cpu-request 100
 slot replica 2
 slot map    1
exit

cdl kafka replica 1
cdl kafka label-config key smi.cisco.com/node-type1
cdl kafka label-config value smf-cdl
cdl kafka external-ip 209.165.201.25 10092 >> Rack-1 external CDL IP
```

Rack-2/Site-2:

```
cdl system-id          2
cdl node-type          session
cdl enable-geo-replication true
cdl zookeeper replica 1

cdl remote-site 1
 db-endpoint host 209.165.201.25 >> Rack-1 external CDL IP
 db-endpoint port 8882
 kafka-server 209.165.201.25 10092 >> Rack-1 external CDL IP
 exit
exit

cdl label-config session
 endpoint key smi.cisco.com/node-type12
 endpoint value smf-cdl
 slot map 1
  key   smi.cisco.com/node-type12
  value smf-cdl
 exit
 index map 1
  key   smi.cisco.com/node-type12
  value smf-cdl
 exit
exit

cdl datastore session
 label-config    session
 geo-remote-site [ 1 ]
 slice-names     [ 1 2 ]
 endpoint cpu-request 100
 endpoint replica 2
 endpoint external-ip 209.165.201.21 >> Rack-2 external CDL IP
 endpoint external-port 8882
 index cpu-request 100
 index replica 2
 index map   1
 slot cpu-request 100
 slot replica 2
 slot map    1
exit

cdl kafka replica 1
cdl kafka label-config key smi.cisco.com/node-type12
cdl kafka label-config value smf-cdl
cdl kafka external-ip 209.165.201.21 10092 >> Rack-2 external CDL IP
```

Lawful Intercept

The Lawful Intercept (LI) feature enables law enforcement agencies (LEAs) to intercept subscriber communications. The LI functionality provides the network operator the capability to intercept control and data messages of the targeted mobile users. To invoke this support, the LEA requests the network operator to start the interception of a particular mobile user. Legal approvals support this request.

1. Lawful Intercept (LI) tap should be configured/enabled on all the sites. If LI configuration fails on one site, LEA should reconfigure it so that for a given subscriber tap is enabled on all the sites.

Note LI tap configuration is not synchronized across sites.

Hence, LI tap configuration is mandatory on all the sites.

For more information on LI tap configuration, contact your Cisco Technical Representative.

2. GR instance awareness is applicable for lawful-intercept src-address only.

   Example:

   ```
   lawful-intercept instance 1 src-addr 209.165.200.225
   ```

   OR

   ```
   lawful-intercept
     instance 1
       src-addr 209.165.200.225
   ```

3. `show` commands are not instance-aware. It shows all the taps configured in a given cluster.

   For more information on LI `show` commands, contact your Cisco Technical Representative.

4. In case all GR instances are in Standby state in a cluster and active LI tap fails with CLI message `Rack is in standby mode, Active Tap is not allowed. Try camp on`, configure camp-on tap for the same subscriber.

RADIUS Configuration

NAS-IP and NAS-Identifier is instance-aware. You can configure different NAS-IP and NAS-Identifier per instance-id in profile-radius configuration. Existing non-instance based NAS-IP and NAS-Identifier configuration is used as default nas-ip and default nas-id for local-instance of the site.

Example

Following are a few configuration examples.

```
profile radius
 attribute
  instance 1
   nas-ip 209.165.200.225  --> Instance-1 specific NAS-IP, used for common AUTH & ACCT
   nas-identifier  smf1    --> Instance-1 specific NAS-Identifier, used for common AUTH &
ACCT
  exit
  instance 2
   nas-ip 209.165.200.230  --> Instance-2 specific NAS-IP, used for common AUTH & ACCT
   nas-identifier  smf2    --> Instance-2 specific NAS-Identifier, used for common AUTH &
```

```
ACCT
  exit
 exit
 accounting
  attribute
   instance 1
    nas-ip 209.165.200.225  --> Instance-1 specific NAS-IP, used for common ACCT
    nas-identifier  smf1    --> Instance-1 specific NAS-Identifier , used for common ACCT
   exit
   instance 2
    nas-ip 209.165.200.230  --> Instance-2 specific NAS-IP, used for common ACCT
    nas-identifier  smf2    --> Instance-2 specific NAS-Identifier , used for common ACCT
   exit
  exit
 exit
 server-group g1
  attribute
   instance 1
    nas-ip 209.165.200.225  --> Instance-1 specific NAS-IP, used for server-group <g1> AUTH
 & ACCT
    nas-identifier  smf1    --> Instance-1 specific NAS-ID, used for server-group <g1> Auth
 &Acct
   exit
   instance 2
    nas-ip 209.165.200.230  --> Instance-2 specific NAS-IP, used for server-group <g1> AUTH
 & ACCT
    nas-identifier  smf2    --> Instance-2 specific NAS-ID,used for server-group <g1>AUTH&ACCT

   exit
  exit
  accounting
   attribute
    instance 1
     nas-ip 209.165.200.225 --> Instance-1 specific NAS-IP, used for server-group <g1> ACCT

     nas-identifier  smf1   --> Instance-1 specific NAS-ID, used for server-group <g1> ACCT

    exit
    instance 2
     nas-ip 209.165.200.230 --> Instance-2 specific NAS-IP, used for server-group <g1> ACCT

     nas-identifier  smf2   --> Instance-2 specific NAS-ID, used for server-group <g1> ACCT

    exit
   exit
  exit
 exit
exit
```

Since **endpoint** *pod* configuration is moved under specific instance, Radius Disconnect-Request VIP is also instance-aware.

```
instance instance-id 1
 endpoint radius
  replicas 1
  interface coa-nas
   vip-ip 209.165.202.130 vip-port 3799  --> Instance-1 specific Radius-Disconnect-Msg-VIP
 & PORT
  exit
 exit
exit
instance instance-id 2
 endpoint radius
  replicas 1
  interface coa-nas
```

```
   vip-ip 209.165.202.129 vip-port 3799  --> Instance-2 specific Radius-Disconnect-Msg-VIP
 & PORT
  exit
 exit
exit
```

Software Upgrade on GR Pairs

Considering `config commit` as reference. The same checklist is also applicable for other upgrade scenarios.

Checklist

Note Don't perform `cluster sync` on both sites (Rack-1/Site-1 and Rack-2/Site-2) at the same time. Trigger manual switchover on Rack-1 before proceeding with Rack-1/Site-1 upgrade.

- Don't peform `config commits` on both sites at the same time. Perform `config commit` on each site separately.
- Prior to `config commit` on Rack-1/Site-1, initiate CLI based switchover on Rack-1/Site-1 and make sure that Rack-2/Site-2 is having Primary ownership for both the instances (instance-id 1 and instance-id 2).
- Perform `config commit` on Rack-1/Site-1. Wait till `config commit` is successful and PODs restart and are back in running state to fetch latest helm charts (if applicable).
- Revert the role of Rack-1/Site-1 to be Primary (Switch/Reset roles on both sites).
- Verify that roles of Rack-1//Site-2 (Primary) and Rack-2//Site-2 (Standby) are as expected.
- Repeat the above checklist for Rack-2/Site-2.

Software Upgrade

Rack-1/Site-1 Upgrade when GR is Enabled

1. Verify that roles of both instances on Rack-1//Site-1 are in PRIMARY/STANDBY.

   ```
   show role instance-id 1
   result "PRIMARY"
   ```

   ```
   show role instance-id 2
   result "STANDBY"
   ```

2. Initiate switch role for both instances on Rack-1/Site-1 to STANDBY with failback-interval of 30 seconds. This step transitions the roles from PRIMARY/STANDBY to STANDBY_ERROR/STANDBY_ERROR.

Note Heartbeat between both the sites should be successful.

```
geo switch-role instance-id 1 role standby failback-interval 30
geo switch-role instance-id 2 role standby failback-interval 30
```

3. Verify that roles of both instances has moved to STANDBY_ERROR on Rack-1/Site-1.

```
show role instance-id 1
result "STANDBY_ERROR"

show role instance-id 2
result "STANDBY_ERROR"
```

4. Verify that roles of both instances has moved to PRIMARY on Rack-2/Site-2.

```
show role instance-id 1
result "PRIMARY"

show role instance-id 2
result "PRIMARY"
```

5. Perform rolling upgrade (or) non-graceful upgrade using system mode shutdown/running as per the requirement on Rack-1/Site-1.
6. Perform the following steps post completion of upgrade procedure. Perform health check on Rack-1/Site-1 and ensure the PODs have come up and Rack-1/Site-1 is healthy.
7. Verify that roles of both instances remain in STANDBY_ERROR mode on Rack-1/Site-1.

```
show role instance-id 1
result "STANDBY_ERROR"

show role instance-id 2
result "STANDBY_ERROR"
```

8. Initiate reset role for both instances on Rack-1/Site-1 to STANDBY. This step transitions the roles from STANDBY_ERROR/STANDBY_ERROR to STANDBY/STANDBY.

```
geo reset-role instance-id 1 role standby
geo reset-role instance-id 2 role standby
```

9. Verify that the roles of both instances have moved to STANDBY on Rack-1/Site-1.

```
show role instance-id 1
result "STANDBY"

show role instance-id 2
result "STANDBY"
```

10. Initiate switch role for instance-id 1 on Rack-2/Site-2 to STANDBY. This step transitions roles of Rack-2/Site-2 from PRIMARY/PRIMARY to STANDBY_ERROR/PRIMARY and Rack-1/Site-1 from STANDBY/STANDBY to PRIMARY/STANDBY.

```
geo switch-role instance-id 1 role standby failback-interval 30
```

11. Verify that roles of the instances on Rack-2/Site-2 are in STANDBY_ERROR/PRIMARY.

```
show role instance-id 1
result "STANDBY_ERROR"

show role instance-id 2
result "PRIMARY"
```

12. Verify that the roles of both instances on Rack-1/Site-1 are in PRIMARY/STANDBY.

```
show role instance-id 1
 result "PRIMARY"

show role instance-id 2
 result "STANDBY"
```

13. Initiate reset role for instance-id 1 on Rack-2/Site-2 to STANDBY. This step transitions the roles of Rack-2/Site-2 from STANDBY_ERROR/PRIMARY to STANDBY/PRIMARY.

```
geo reset-role instance-id 1 role standby
```

14. Verify that roles of both instances on Rack-2/Site-2 are in STANDBY/PRIMARY.

```
show role instance-id 1
result "STANDBY"

show role instance-id 2
 result "PRIMARY"
```

Rack-2/Site-2 Upgrade when GR is Enabled

1. Verify that roles of both instances on Rack-2/Site-2 are in STANDBY/PRIMARY.

```
show role instance-id 1
result "STANDBY"

show role instance-id 2
result "PRIMARY"
```

2. Initiate switch role for both instances on Rack-2/Site-2 to STANDBY with failback-interval of 30 seconds. This step transitions the roles from STANDBY/PRIMARY to STANDBY_ERROR/STANDBY_ERROR.

```
geo switch-role instance-id 1 role standby failback-interval 30
geo switch-role instance-id 2 role standby failback-interval 30
```

3. Verify that roles of both instances move to STANDBY_ERROR on Rack-2/Site-2.

```
show role instance-id 1
result "STANDBY_ERROR"

show role instance-id 2
result "STANDBY_ERROR"
```

4. Verify that roles of both instances move to PRIMARY on Rack-1/Site-1.

```
show role instance-id 1
 result "PRIMARY"

show role instance-id 2
 result "PRIMARY"
```

5. Perform rolling upgrade (or) non-graceful upgrade via system mode shutdown/running as per the requirement on Rack-2/Site-2.

6. Perform the subsequent steps post completion of upgrade procedure. Perform health check on Rack-2/Site-2 and ensure the PODs have come up and Rack-2/Site-2 is healthy

7. Verify that roles of both the instances remain in STANDBY_ERROR on Rack-2/Site-2.

```
show role instance-id 1
result "STANDBY_ERROR"

show role instance-id 2
result "STANDBY_ERROR"
```

8. Initiate reset role for both instances on Rack-2/Site-2 to STANDBY. This step transitions the roles from STANDBY_ERROR/STANDBY_ERROR to STANDBY/STANDBY.

```
geo reset-role instance-id 1 role standby
geo reset-role instance-id 2 role standby
```

9. Verify that the roles of both instances move to STANDBY on Rack-2/Site-2.

```
show role instance-id 1
result "STANDBY"
```

```
show role instance-id 2
result "STANDBY"
```

10. Initiate switch role for instance-id 2 on Rack-1/Site-1 to STANDBY. This step transitions roles of Rack-1/Site-2 from PRIMARY/PRIMARY to PRIMARY/STANDBY_ERROR and Rack-2/Site-2 from STANDBY/STANDBY to STANDBY/PRIMARY.

    ```
    geo switch-role instance-id 2 role standby failback-interval 30
    ```

11. Verify that roles of both instances on Rack-1/Site-1 are in PRIMARY/STANDBY_ERROR.

    ```
    show role instance-id 1
    result "PRIMARY"

    show role instance-id 2
    result "STANDBY_ERROR"
    ```

12. Verify that roles of both instances on Rack-2/Site-2 are in STANDBY/PRIMARY.

    ```
    show role instance-id 1
    result "STANDBY"

    show role instance-id 2
    result "PRIMARY"
    ```

13. Initiate reset role for instance-id 2 on Rack-1/Site-1 to STANDBY. This step transitions the roles of Rack-1/Site-1 from PRIMARY/STANDBY_ERROR to PRIMARY/STANDBY.

    ```
    geo reset-role instance-id 2 role standby
    ```

14. Verify that roles of both the instances on Rack-1/Site-1 are in PRIMARY/STANDBY.

    ```
    show role instance-id 1
    result "PRIMARY"

    show role instance-id 2
    result "STANDBY"
    ```

Manual CLI Switchover

The following section provides information on manual CLI based switchover commands.

Geo Switch Role

To switch GR role (for example, role Primary to Standby), use the following command.

geo switch-role role *role* **instance-id** *gr_instanceId* **failback-interval** *interval_in_sec*

NOTES:

- **role** *role*—Specify new role for the given site.

  Role can be maintenance, primary, or standby (maintenance mode is not supported).

- **instance-id** *gr_instanceId*—Specify GR Instance ID

- **failback-interval** *interval_in_sec*—Specify interval in seconds between notify failover and actual failover.

`geo switch-role` command triggers manual failover from one site to another site for specific instance ID. The site which triggers the failover is moved from PRIMARY role to STANDBY_ERROR role. In between,

the site which triggers failover, sends a failover (trigger GR) message to another site. The other site which receives the failover message is moved from STANDBY role to PRIMARY role.

Geo Reset Role

To reset the GR instance role (for example, role from STANDBY to PRIMARY), use the following command:

geo reset-role role *role* **instance-id** *gr_instanceId*

NOTES:

- **role** *role*—Specify new role for the given site.

  Role can be maintenance, primary, or standby (maintenance mode is not supported).

- **instance-id** *gr_instanceId*—Specify GR Instance ID.

`geo reset-role` command triggers change in the role for the given instance on local site. Remote site will not receive any message for the same command. It is only possible to change the role for the given instance ID from STANDBY_ERROR to STANDBY and STANDBY to PRIMARY. Another role change is not possible.

Troubleshooting

This section describes about various applicable troubleshooting scenarios.

show/clear Commands

This section describes show/clear commands that help in debugging issues.

clear subscriber

To clear gr-instance aware subscriber, use the following command:

clear subscriber all gr-instance *gr_instanceId*

Note **gr-instance** is optional parameter. If **gr-instance** is not specified, `show subscriber all` considers the local instance-id of that rack/site.

Example

The following is a configuration example.

```
clear subscriber all gr-instance 1
result
ClearSubscriber Request submitted
```

show BFD Status

To view the BFD status of neighbors, use the following command:

show bfd-neighbor

Example

The following is a list of few configuration examples:

```
show bfd-neighbor
status-details

-----example-bgp-ep-1 ----

Peer             Status

 209.165.202.142  STATE_DOWN
-----example-bgp-ep-2 ----

Peer             Status

 209.165.202.142  STATE_DOWN
```

```
show bfd-neigbor
status-details

-----bgpspeaker-pod-1 ----

Peer             Status

 209.165.202.131
-----bgpspeaker-pod-2 ----

Peer             Status

 209.165.202.131   STATE_UP
```

show BGP Global

To view BGP global configuration, use the following command:

`show bgp-global`

Example

The following is a list of few configuration examples:

```
show bgp-global
global-details
-----example-bgp-ep-2 ----
AS:        65000
Router-ID: 209.165.202.149
Listening Port: 179, Addresses: 209.165.202.149
-----example-bgp-ep-1 ----
AS:        65000
Router-ID: 209.165.202.148
Listening Port: 179, Addresses: 209.165.202.148
```

```
show bgp-global
global-details

-----bgpspeaker-pod-2 ----
AS:        65061
Router-ID: 209.165.202.132
Listening Port: 179, Addresses: 209.165.202.132
```

show bgp kernel route

To view BGP kernel configured routes, use the following command:

show bgp-kernel-route kernel-route

Example

The following is a list of few configuration examples:

```
show bgp-kernel-route
kernel-route

-----example-bgp-ep-2 ----

 DestinationIP  SourceIP        Gateway

-----example-bgp-ep-1 ----

 DestinationIP      SourceIP          Gateway
 209.165.202.133    209.165.202.148   209.165.202.142
 209.165.202.134    209.165.202.148   209.165.202.142
```

```
show bgp-kernel-route
kernel-route

-----bgpspeaker-pod-2 ----

 DestinationIP        SourceIP            Gateway

 209.165.202.135    209.165.202.132    209.165.202.131

-----bgpspeaker-pod-1 ----

 DestinationIP  SourceIP        Gateway
```

show bgp neighbors

To view BGP neighbors status, use the following command

show bgp-neighbors neighbor-details
show bgp-neighbors ip *ip_address* **neighbor-details**

Example

The following is a list of few configuration examples:

```
show bgp-neighbors neighbor-details
-----example-bgp-ep-1 ----
Peer             AS  Up/Down State       |#Received  Accepted
209.165.202.142 60000 00:25:06 Establ      |        3         3
-----example-bgp-ep-2 ----
Peer             AS Up/Down State       |#Received  Accepted
209.165.202.142 60000   never Idle        |        0         0
```

```
show bgp-neighbors ip 209.165.202.142 neighbor-details
-----example-bgp-ep-2 ----
BGP neighbor is 209.165.202.142, remote AS 60000
  BGP version 4, remote router ID unknown
  BGP state = ACTIVE
  BGP OutQ = 0, Flops = 0
  Hold time is 0, keepalive interval is 0 seconds
  Configured hold time is 90, keepalive interval is 30 seconds

  Neighbor capabilities:
    multiprotocol:
        ipv4-unicast:   advertised
    route-refresh:      advertised
```

```
    extended-nexthop:   advertised
        Local:  nlri: ipv4-unicast, nexthop: ipv6
    4-octet-as: advertised
  Message statistics:
                          Sent       Rcvd
    Opens:                 130          0
    Notifications:           0          0
    Updates:                 0          0
    Keepalives:              0          0
    Route Refresh:           0          0
    Discarded:               0          0
    Total:                 130          0
  Route statistics:
    Advertised:              0
    Received:                0
    Accepted:                0

-----example-bgp-ep-1 ----
BGP neighbor is 209.165.202.142, remote AS 60000
  BGP version 4, remote router ID 209.165.202.136
  BGP state = ESTABLISHED, up for 00:25:20
  BGP OutQ = 0, Flops = 0
  Hold time is 90, keepalive interval is 30 seconds
  Configured hold time is 90, keepalive interval is 30 seconds

  Neighbor capabilities:
    multiprotocol:
        ipv4-unicast:   advertised and received
    route-refresh:      advertised and received
    extended-nexthop:   advertised
        Local:  nlri: ipv4-unicast, nexthop: ipv6
    4-octet-as: advertised and received
  Message statistics:
                          Sent       Rcvd
    Opens:                   1          1
    Notifications:           0          0
    Updates:                 1          1
    Keepalives:             51         51
    Route Refresh:           0          0
    Discarded:               0          0
    Total:                  53         53
  Route statistics:
    Advertised:              0
    Received:                3
    Accepted:                3
```

show bgp route summary

To view BGP route summary, use the following command:

show bgp-route-summary

Example

The following is a configuration example.

```
show bgp-route-summary
route-details
-----example-bgp-ep-1 ----
Table afi:AFI_IP safi:SAFI_UNICAST
Destination: 5, Path: 5
-----example-bgp-ep-2 ----
Table afi:AFI_IP safi:SAFI_UNICAST
Destination: 2, Path: 2
```

show BGP Routes

To view BGP routes information, use the following command:

show bgp-routes

Example

The following is a configuration example:

```
show bgp-routes
bgp-route

-----example-bgp-ep-1 ----
   Network              Next Hop             AS_PATH      Age        Attrs
*> 209.165.202.133/24   209.165.202.142        60000      00:25:55   [{Origin: i} {Med: 0}]
*> 209.165.200.225/32   209.165.202.148                  00:26:00   [{Origin: e} {LocalPref:
 100} {Med: 600}]
*> 209.165.202.134/24   209.165.202.142        60000      00:25:55   [{Origin: i} {Med: 0}]
*> 209.165.202.140/24   209.165.202.142        60000      00:25:55   [{Origin: i} {Med: 0}]
*> 209.165.202.146/32   209.165.202.148                  00:26:00   [{Origin: e} {LocalPref:
 100} {Med: 600}]

-----example-bgp-ep-2 ----
   Network              Next Hop             AS_PATH      Age        Attrs
*> 209.165.200.225/32   209.165.202.149                  00:26:24   [{Origin: e} {LocalPref:
 100} {Med: 600}]
*> 209.165.202.146/32   209.165.202.149                  00:26:24   [{Origin: e} {LocalPref:
 100} {Med: 600}]
```

show endpoint

To view endpoints that are now gr-instance aware, use the following command:

show endpoint all grInstance *gr_instanceId*

Note **grInstance** is optional parameter. If **grInstance** is not specified, `show subscriber all` considers the local instance-id of that rack/site.

Example

The following is a configuration example:

```
show endpoint all grInstance 1
                                                                                START
STOPPED  GR
ENDPOINT                          ADDRESS        TYPE  STATUS   INTERFACE  INTERNAL  TIME
     TIME     INSTANCE
-------------------------------------------------------------------------------------------------
209.165.202.137:2123       209.165.202.137:2123  Udp   Started             false     10
hours  <none>   1
Gtpu:209.165.202.137:2152  209.165.202.137:2152  Udp   Started  GTPU       false     10
hours  <none>   1
N4:209.165.202.137:8806    209.165.202.137:8806  Udp   Started  N4         false     10
hours  <none>   1
S2B-GTP                    209.165.202.138:2124  Udp   Started  s2b        false     10
hours  <none>   1
S5-GTP                     209.165.202.138:2125  Udp   Started  s5         false     10
hours  <none>   1
S5S8S2B-GTP                209.165.202.138:2123  Udp   Started  s5s8s2b    false     10
```

```
hours  <none>   1
Sxa:209.165.202.137:8805   209.165.202.137:8805  Udp   Started  SXA        false     10
hours  <none>   1
n10-1                      209.165.202.139:9010  Rest  Started  N10-1      false     10
hours  <none>   1
n11-1                      209.165.202.139:9011  Rest  Started  N11-1      false     10
hours  <none>   1
n40-1                      209.165.202.139:9040  Rest  Started  N40-1      false     10
hours  <none>   1
n7-1                       209.165.202.139:9007  Rest  Started  N7-1       false     10
hours  <none>   1
sbi-1                      209.165.202.139:8090  Rest  Started  SBI-1      false     10
hours  <none>   1
```

show ETCD/Cache Pod Replication

To view replication details for etcd and cache-pod data, use the following command:

show georeplication checksum instance-id *gr_instanceId*

Example

The following is a configuration example:

```
show georeplication checksum instance-id
Value for 'instance-id' (<string>): 1
checksum-details
--              ----    --------
 ID             Type    Checksum
 --             ----    --------
 1              ETCD    1617984439
 IPAM           CACHE   1617984439
 NRFCache       CACHE   1617984439
 NRFSubs        CACHE   1617984439
 IDMGR          CACHE   1617984439
 NRFMgmt        CACHE   1617984439
```

show geo role

To view the current role of GR instance, use the following command:

show role instance-id *gr_instanceId*

Note The following is a list of possible values for the role:

- PRIMARY
- STANDBY
- FAILOVER_INIT
- FAILOVER_COMPLETE
- STANDBY_ERROR
- FAILBACK_STARTED

Example

The following is a list of few configuration examples:

```
show role instance-id 1
result
"PRIMARY"

show role instance-id 2
result
"STANDBY"
```

show ipam dp with type and address

To view the instance ID and flag to indicate chunk for remote instance, use the following command:

show ipam dp { *dp_type* } { *addr_type* }

NOTES:

- **dp** *dp_type*—Specify DP type.
- *addr_type*—Specify IPv4/IPv6 address type.

Example

The following is a configuration example.

```
show ipam dp 209.165.202.145:209.165.202.144  ipv4-addr
================================================================
Flag Indication: S(Static) O(Offline) R(For Remote Instance)
G:N/P Indication: G(GR InstId) N(Native NM InstId) P(Peer NM InstId)
================================================================
StartAddress    EndAddress      AllocContext                    Route              G:N/P
Utilization Flag
================================================================
209.165.200.240 209.165.200.243 209.165.202.145:209.165.202.144 209.165.200.240/24 1:0/1
0.00%        R
================================================================
```

show ipam dp

To view all the instances this DP has chunks from, use the following command:

show ipam dp *dp_name*

NOTES:

- **dp** *dp_name*—Specify data plane allocation name.

Example

The following is a configuration example.

```
show ipam dp 209.165.202.145:209.165.202.144
------------------------------------------------------
Ipv4Addr [Total/Used/Utilization] = 257 / 1 / 0.39%
Ipv6Addr [Total/Used/Utilization] = 0 / 0 / 0.00%
Ipv6Prefix [Total/Used/Utilization] = 2048 / 0 / 0.00%
Instance ID = 1
------------------------------------------------------
```

show ipam pool

To view instance ID information under which pool is configured, use the following command:

show ipam pool *pool_name*

NOTES:

- **pool** *pool_name*—Specify pool name.

Example

The following is a list of few configuration examples.

```
show ipam pool

=============================================================================================
PoolName                        Ipv4Utilization  Ipv6AddrUtilization  Ipv6PrefixUtilization
=============================================================================================
poolv6DNN2                      0.00%            0.00%                0.00%
poolv6                          0.00%            0.00%                0.00%
poolv4vDNN                      0.00%            0.00%                0.00%
poolv4DNN2                      0.00%            0.00%                0.00%
poolv4                          0.00%            0.00%                0.00%
poolv6vDNN                      0.00%            0.00%                0.00%
poolv4DNN3                      -                -                    -
=============================================================================================

show ipam pool poolv4DNN3
-----------------------------------------------------------
Ipv4Addr [Total/Used/Utilization] = 2814 / 0 / -
Ipv6Addr [Total/Used/Utilization] = 0 / 0 / -
Ipv6Prefix [Total/Used/Utilization] = 65536 / 0 / -
Instance ID = 1
isStatic = true
-----------------------------------------------------------

show ipam pool poolv4
-----------------------------------------------------------
 Ipv4Addr   [Total/Used/Utilization] = 2814 / 0 / 0.00%
 Ipv6Addr   [Total/Used/Utilization] = 0 / 0 / 0.00%
 Ipv6Prefix [Total/Used/Utilization] = 0 / 0 / 0.00%
 Instance ID                         = 1
-----------------------------------------------------------
```

show nrf discovery-info discovery-filter

To view GR Instance ID information to determine for which GR instance the discovery filter information belongs, use the following command:

show nrf discovery-info *nf_type* **discovery-filter**

Example

The following is a configuration example.

```
=======================================================
-------------------------------------------------------------
Discovery Filter: dnn=intershat;
Expiry Time: 1580146356
GR Instance ID: 1
-------------------------------------------------------------
=====================================================================
```

show nrf discovery-info

To view GR Instance ID information to determine for which GR instance the discovery information belongs, use the following command:

show nrf discovery-info

Example

The following is a configuration example.

```
show nrf discovery-info
=================================================
------Discovered NFs:-------
    NF Type: AMF
    Number of Discovery Filters: 15
    Number of NF Profiles: 15
    GR Instance ID: 1
------Discovered NFs:-------
    NF Type: UDM
    Number of Discovery Filters: 1
    Number of NF Profiles: 3
    GR Instance ID: 2
=================================================
```

show nrf registration-info

To view GR Instance ID information to determine which GR instance the registration information belongs to, use the following command:

show nrf registration-info

Example

The following is a configuration example.

```
show nrf registration-info
=========================================================================
 NF Status: Not Registered
 Registration Time:
 Active MgmtEP Name:
 Heartbeat Duration: 0
 GR Instance ID: 1
========

show nrf registration-info

=========================================================================
 Gr-instance:
 NF Status: Not Registered
 Registration Time:
 Active MgmtEP Name:
 Heartbeat Duration: 0
 Uri:
 Host Type:

=========================================================================
 Gr-instance:
 NF Status: Not Registered
 Registration Time:
 Active MgmtEP Name:
 Heartbeat Duration: 0
 Uri:
```

```
 Host Type:

=====================================================================
```

show nrf subscription-info

To view GR Instance ID information to determine for which GR instance the subscription information belongs, use the following command:

show nrf subscription-info

Example

The following is a configuration example.

```
show nrf subscription-info
=====================================================================
NF Instance Id: f9882966-a253-32d1-8b82-c785b34a7cc9
SubscriptionID : subs123459
Actual Validity Time : 2020-01-21 12:39:45 +0000 UTC
Requested Validity Time : 2020-01-21 12:39:45 +0000 UTC
GR Instance ID: 1
=====================================================================
```

show peers

To view peers that are now gr-instance aware, use the following command:

show peers all grInstance *gr_instanceId*

Note **grInstance** is optional parameter. If **grInstance** is not specified, `show subscriber all` considers the local instance-id of that rack/site.

Example

The following is a configuration example.

```
show peers all grInstance 1
                                                          POD              CONNECTED
ADDITIONAL  INTERFACE  GR
ENDPOINT  LOCAL ADDRESS   PEER ADDRESS     DIRECTION  INSTANCE TYPE TIME     RPC DETAILS
NAME       INSTANCE
-------------------------------------------------------------------------------------------------
<none>  209.165.202.139 209.165.201.22:8001 Outbound rest-ep-0 Rest 10 hours UDM <none>
n10        1
<none>  209.165.202.139 209.165.201.22:8002 Outbound rest-ep-0 Rest 10 hours AMF <none>
n11        1
<none>  209.165.202.139 209.165.201.22:8003 Outbound rest-ep-0 Rest 10 hours PCF <none>
n7         1
<none>  209.165.202.139 209.165.201.22:8004 Outbound rest-ep-0 Rest 10 hours CHF <none>
n40        1
<none>  209.165.202.139 209.165.201.22:9040 Outbound rest-ep-0 Rest 10 hours CHF <none>
n40
```

show role

To view the instance role, use the following command:

show role

Example

The following is a list of few configuration examples:

```
show role instance-id 2
result "PRIMARY"

show role instance-id 1
result "PRIMARY"
```

show subscriber

To view subscriber details that are made gr-instance aware, use the following command:

show subscriber { all | gr-instance *gr_instanceId* **}**

Note `show subscriber all` displays only the local instance subscriber details.

gr-instance is optional parameter. If **gr-instance** is not specified, `show subscriber all` considers the local instance-id of that rack/site.

Example

The following is a configuration example.

```
show subscriber gr-instance 1 all
subscriber-details
{
"subResponses": [
[
""
],
[
""
],
[
"roaming-status:homer",
"supi:imsi-123456789300001",
"gpsi:msisdn-22331010301010",
"psid:1",
"dnn:intershat",
"emergency:false",
"rat:nr",
"access:3gpp access",
"connectivity:5g",
"udm-uecm:209.165.202.150",
"udm-sdm:209.165.202.150",
"auth-status:unauthenticated",
"pcfGroupId:PCF-*",
"policy:2",
"pcf:209.165.202.152",
"upf:209.165.202.154",
"upfEpKey:209.165.202.154:209.165.202.158",
"ipv4-addr:v4pool1/209.165.200.250",
"ipv4-pool:v4pool1",
"ipv4-range:v4pool1/209.165.200.249",
"ipv4-startrange:v4pool1/209.165.200.250",
"id-index:1:0:0:32768",
"id-value:8",
"chfGroupId:CHF-*",
"chf:209.165.202.151",
```

```
"amf:209.165.202.153",
"peerGtpuEpKey:209.165.202.154:209.165.202.155",
"namespace:smf",
"nf-service:smf"
]
]
}
```

Monitor Subscriber

To capture messages for subscriber (gr-instance aware), use the following command:

monitor subscriber [supi] [imsi] [imei] (capture-duration) (internal-messages) (transaction-logs) (nf-service) (gr-instance)

Note In 2021.02 and later releases, the **namespace** keyword is deprecated and replaced with the **nf-service** keyword.

NOTES:

- **supi** —Specify the subscriber identifier.

  Example: imsi-123456789, imsi-123*

- **imsi** —Specify the IMSI value.

  Example: 123456789, *

- **imei** —Specify the IMEI value.
- Example: 123456789012345, *
- **capture-duration** —(Optional) Used to specify the duration in seconds during which monitor subscriber is enabled. Default value is 300 secs.
- **internal-messages** —(Optional) When set to yes, it enables internal messaging. By default, it is disabled.
- **transaction-logs** —(Optional) When set to yes, it enables transaction logging. By default, it is disabled.

Note Messages and transaction logs are mutually exclusive.

- **namespace** —Deprecated option. Use nf-service instead.
- **nf-service**—(Optional) Specify the NF service. Possible values are sgw, smf. Default value is none.
- **gr-instance**—(Optional) Monitor subscriber for a given gr-instance only.

Example

The following is a configuration example.

```
monitor subscriber imsi 123456789 gr-instance 1
supi: imsi-123456789
captureDuration: 300
enableInternalMsg: false
enableTxnLog: false
```

```
namespace(deprecated. Use nf-service instead.): none
nf-service: none
gr-instance: 1
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   295  100    98  100   197  10888  21888 --:--:-- --:--:-- --:--:-- 29500
Command: --header Content-type:application/json --request POST --data
{"commandname":"mon_sub","parameters":{"supi":"imsi-123456789","duration":300,
"enableTxnLog":false,"enableInternalMsg":false,"action":"start","namespace":"none",
"nf-service":"none","grInstance":1}} http://oam-pod:8879/commands
Result start mon_sub, fileName
->logs/monsublogs/none.imsi-123456789_TS_2021-04-09T09:59:59.964148895.txt
Starting to tail the monsub messages from file:
logs/monsublogs/none.imsi-123456789_TS_2021-04-09T09:59:59.964148895.txt
Defaulting container name to oam-pod.
Use 'kubectl describe pod/oam-pod-0 -n smf' to see all the containers in this pod.
```

For more information on Monitor Protocol on cnSGW-C, see the *Monitor Subscriber and Monitor Protocol Support* chapter in the *UCC Serving Gateway Control Plane Function - Configuration and Administration Guide*.

Monitor Protocol

To capture packets on different interfaces (gr-instance aware), use the following command:

monitor protocol [interface] (capture-duration) (count) (level) (gr-instance)

NOTES:

- **interface**—Interface on which PCAP is captured.

  Example: sbi, pfcp, gtpu, gtpc, gtp, radius

- **list**—Monitor protocol list files.
- **capture-duration**—(Optional) Used to specify the duration in secs during which PCAP is captured. Default value is 300 secs.
- **pcap**—(Optional) When set to yes, it enables PCAP file generation. By default, the value is "no" (disabled).
- **gr-instance**—(Optional) Monitor subscriber for a given gr-instance only.

Example

The following is a configuration example.

```
monitor protocol interface sbi gr-instance 1
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   220  100    95  100   125   8636  11363 --:--:-- --:--:-- --:--:-- 20000
Command: --header Content-type:application/json --request POST --data
{"commandname":"mon_pro","parameters":{"interface":"sbi","duration":300,"action":
"start","enable_pcap":false,"grInstance":1}} http://oam-pod:8879/commands
Result start mon_pro, fileName
->logs/monprologs/sessintfname_sbi_at_2021-04-30T05:26:22.712229347.txt
Starting to tail the monpro messages from file:
logs/monprologs/sessintfname_sbi_at_2021-04-30T05:26:22.712229347.txt
Defaulting container name to oam-pod.
Use 'kubectl describe pod/oam-pod-0 -n cn' to see all of the containers in this pod.
```

For more information on Monitor Protocol on cnSGW-C, see the *Monitor Subscriber and Monitor Protocol Support* chapter in the *UCC Serving Gateway Control Plane Function - Configuration and Administration Guide*.

Geographic Redundancy OAM Support

This section describes operations, administration, and maintenance information for this feature.

Health Check

The following section provides information on GR setup health check.

- All critical pods are in good condition to serve user traffic.

  Use the following command to check whether GR and CDL related pods are in Running state.

  ```
  kubectl get pods -n cn-cn1 -o wide | grep georeplication-pod
  kubectl get pods -n cn-cn1 -o wide | grep cdl
  kubectl get pods -n cn-cn1 -o wide | grep mirror-maker
  ```

- Keepalived pods are in healthy state to monitor all VIPs which are configured for check-interface/check-port.

  Use the following command to check whether keepalived pods in “smi-vips” namespace are in “Running” state.

  ```
  kubectl get pods -n smi-vips
  ```

- Health-check of pods related to CDL: Check the status of CDL db-endpoint, slot and indexes. All should be in STARTED or ONLINE state for both System IDs 1 and 2.

  ```
  cdl show status
  message params: {cmd:status mode:cli dbName:session sessionIn:{mapId:0 limit:500 key:
  purgeOnEval:0 filters:[] nextEvalTsStart:0 nextEvalTsEnd:0 allReplicas:false
  maxDataSize:4096} sliceName:}
  db-endpoint {
      endpoint-site {
          system-id 1
          state STARTED
          total-sessions 4
          site-session-count 2
          total-reconciliation 0
          remote-connection-time 66h37m31.36054781s
          remote-connection-last-failure-time 2021-07-13 11:24:10.233825924 +0000 UTC
          slot-geo-replication-delay 2.025396ms
      }
      endpoint-site {
          system-id 2
          state STARTED
          total-sessions 4
          site-session-count 2
          total-reconciliation 0
          remote-connection-time 66h58m49.83449066s
          remote-connection-last-failure-time 2021-07-13 11:02:51.759971655 +0000 UTC
          slot-geo-replication-delay 1.561816ms
      }
  }
  slot {
      map {
          map-id 1
          instance {
  ```

```
                system-id 1
                instance-id 1
                records 4
                capacity 2500000
                state ONLINE
                avg-record-size-bytes 1
                up-time 89h38m37.335813523s
                sync-duration 9.298061ms
            }
            instance {
                system-id 1
                instance-id 2
                records 4
                capacity 2500000
                state ONLINE
                avg-record-size-bytes 1
                up-time 89h39m11.1268024s
                sync-duration 8.852556ms
            }
            instance {
                system-id 2
                instance-id 1
                records 4
                capacity 2500000
                state ONLINE
                avg-record-size-bytes 1
                up-time 89h28m38.274713022s
                sync-duration 8.37766ms
            }
            instance {
                system-id 2
                instance-id 2
                records 4
                capacity 2500000
                state ONLINE
                avg-record-size-bytes 1
                up-time 89h29m37.934345015s
                sync-duration 8.877442ms
            }
        }
    }
    index {
        map {
            map-id 1
            instance {
                system-id 1
                instance-id 1
                records 4
                capacity 60000000
                state ONLINE
                up-time 89h38m16.119032086s
                sync-duration 2.012281769s
                leader false
                geo-replication-delay 10.529821ms
            }
            instance {
                system-id 1
                instance-id 2
                records 4
                capacity 60000000
                state ONLINE
                up-time 89h39m8.47664588s
                sync-duration 2.011171261s
                leader true
```

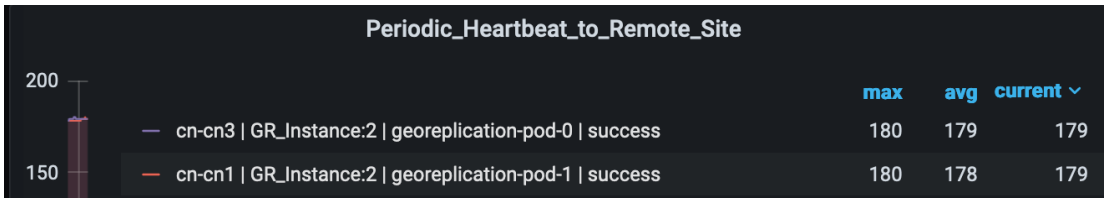
```
            leader-time 89h38m53.761213379s
            geo-replication-delay 10.252683ms
        }
        instance {
            system-id 2
            instance-id 1
            records 4
            capacity 60000000
            state ONLINE
            up-time 89h28m29.5479133s
            sync-duration 2.012101957s
            leader false
            geo-replication-delay 15.974538ms
        }
        instance {
            system-id 2
            instance-id 2
            records 4
            capacity 60000000
            state ONLINE
            up-time 89h29m11.633496562s
            sync-duration 2.011566639s
            leader true
            leader-time 89h28m51.29928233s
            geo-replication-delay 16.213323ms
        }
    }
}
```

- CDL replication status

  Check whether four gRPC connections are established between the CDL EP session pods (of each namespace) across the racks in **GRPC_Connections_to_RemoteSite** panel of **CDL Replication Stats** Grafana dashboard. Check Grafana on both racks.

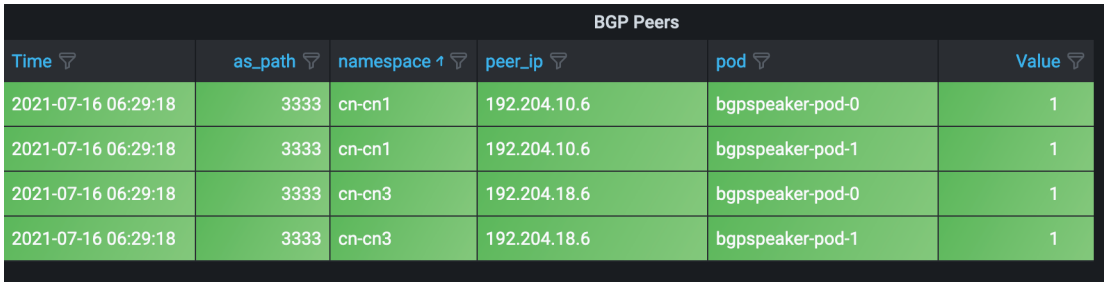
- Admin port status between the racks for geo-replication.

  Check heartbeat messages between geo-replication pods across the racks in **Periodic_Heartbeat_to_Remote_Site** panel of **GR Statistics** Grafana dashboard.

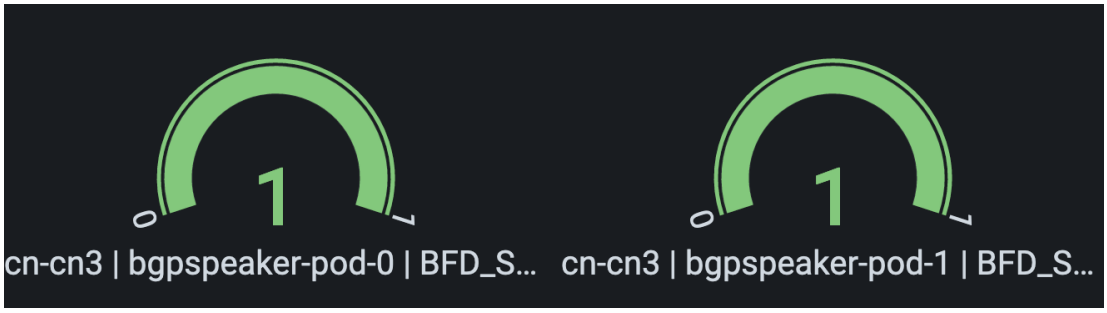


- BGP/BFD link status on rack

  Check whether neighborship with BGP peers is established in **BGP Peers** panel of **BGP, BFD Statistics** Grafan dashboard.



  Check whether BFD link is in connected state in **BFD Link Status** panel of **BGP, BFD Statistics** Grafana dashboard.



- Roles of each instances are in healthy state

  Check that in each rack the roles are not in STANDBY_ERROR state at any point of time.

  - Active/Standby model: Roles should be in the following states on each rack

    Rack-1/Site-1:

    ```
    show role instance-id 1
    result "PRIMARY"
    show role instance-id 2
    result "PRIMARY"
    ```

    Rack-2/Site-2:

    ```
    show role instance-id 1
    result "STANDBY"
    show role instance-id 2
    result "STANDBY"
    ```

  - Active/Active model: Roles should be in the following states on each rack.

    Rack-1/Site-1:

```
show role instance-id 1
result "PRIMARY"
show role instance-id 2
result "STANDBY"
```

Rack-2/Site-2:

```
show role instance-id 1
result "STANDBY"
show role instance-id 2
result "PRIMARY"
```

Recovery Procedure

On Rack-1/Site-1

1. Verify that roles of both instances on Rack-1/Site-1 are in STANDBY_ERROR.

   ```
   show role instance-id 1
   result "STANDBY_ERROR"

   show role instance-id 2
   result "STANDBY_ERROR"
   ```

2. Initiate reset role for both instances on Rack-1/Site-1 to STANDBY. This step transitions the roles from STANDBY_ERROR/STANDBY_ERROR to STANDBY/STANDBY.

   ```
   geo reset-role instance-id 1 role standby
   geo reset-role instance-id 2 role standby
   ```

3. Verify that roles of both instances have moved to STANDBY on Rack-1/Site-1.

   ```
   show role instance-id 1
   result "STANDBY"

   show role instance-id 2
   result "STANDBY"
   ```

4. Initiate switch role for instance-id 1 on Rack-2/Site-2 to STANDBY with failback-interval of 30 seconds. This step transitions the roles of Rack-2/Site-2 from PRIMARY/PRIMARY to STANDBY_ERROR/PRIMARY and Rack-1/Site-1 from STANDBY/STANDBY to PRIMARY/STANDBY.

   ```
   geo switch-role instance-id 1 role standby failback-interval 30
   ```

5. Verify that roles of both instances on Rack-2/Site-2 are in STANDBY_ERROR/PRIMARY.

   ```
   show role instance-id 1
   result "STANDBY_ERROR"

   show role instance-id 2
   result "PRIMARY"
   ```

6. Verify that roles of both instances on Rack-1/Site-1 are in PRIMARY/STANDBY.

   ```
   show role instance-id 1
   result "PRIMARY"

   show role instance-id 2
   result "STANDBY"
   ```

7. Initiate reset role for instance-id 1 on Rack-2/Site-2 to STANDBY. This step transitions the roles of Rack-2/Site-2 from STANDBY_ERROR/PRIMARY to STANDBY/PRIMARY.

   ```
   geo reset-role instance-id 1 role standby
   ```

8. Verify that the roles of Rack-2/Site-2 are in STANDBY/PRIMARY.

```
show role instance-id 1
result "STANDBY"

show role instance-id 2
result "PRIMARY"
```

On Rack-2/Site-2

1. Verify that roles of both the instances on Rack-2/Site-2 are in STANDBY_ERROR.

```
show role instance-id 1
result "STANDBY_ERROR"

show role instance-id 2
result "STANDBY_ERROR"
```

2. Initiate reset role for both instances on Rack-2/Site-2 to STANDBY. This step transitions the roles from STANDBY_ERROR/STANDBY_ERROR to STANDBY/STANDBY.

```
geo reset-role instance-id 1 role standby
geo reset-role instance-id 2 role standby
```

3. Verify that the roles of both the instances move to STANDBY on Rack-2/Site-2.

```
show role instance-id 1
result "STANDBY"

show role instance-id 2
result "STANDBY"
```

4. Initiate switch role for instance-id 2 on Rack-1/Site-1 to STANDBY. This step transitions roles of Rack-1/Site-1 from PRIMARY/PRIMARY to PRIMARY/STANDBY_ERROR and Rack-2/Site-2 from STANDBY/STANDBY to STANDBY/PRIMARY.

```
geo switch-role instance-id 2 role standby failback-interval 30
```

5. Verify that roles of instances on Rack-1/Site-1 are in PRIMARY/STANDBY_ERROR mode.

```
show role instance-id 1
result "PRIMARY"

show role instance-id 2
result "STANDBY_ERROR"
```

6. Verify that roles of instances on Rack-2/Site-2 are in STANDBY/PRIMARY mode.

```
show role instance-id 1
result "STANDBY"

show role instance-id 2
result "PRIMARY"
```

7. Initiate reset role for instance-id 2 on Rack-1/Site-1 to STANDBY. This step transitions the roles of Rack-1/Site-1 from PRIMARY/STANDBY_ERROR to PRIMARY/STANDBY.

```
geo reset-role instance-id 2 role standby
```

8. Verify that roles of instances on Rack-1/Site-1 are in PRIMARY/STANDBY.

```
show role instance-id 1
result "PRIMARY"

show role instance-id 2
result "STANDBY"
```

Key Performance Indicators (KPIs)

The following section describes KPIs.

ETCD/Cachepod Replication KPIs

The following table lists ETCD/Cachepod Replication KPIs.

Table 5: geo_replication_total KPIs

KPI Name	Description	Labels	Possible Values
geo_replication _total	This KPI displays total number of replication requests/responses for various Sync types and Replication types.	ReplicationRequest Type	Request / Response
		ReplicationSync Type	Immediate / Deferred / Pull
		ReplicationNode	ETCD / CACHE_POD / PEER
		ReplicationReceiver	Local / Remote
		status	True / False
		status_code	Error code/description

Geo Rejected Role Change KPIs

The following table lists Geo Rejected Role Change KPIs.

Table 6: Geo Rejected Role Change KPIs

KPI Name	Description	Labels	Possible Values
geo_RejectedRole Changed_total	This KPI displays the total number of rejected requests/calls received for STANDBY instance. After the count, the same instance is moved to PRIMARY.	RejectedCount	Number value indicating rejected calls/requests received for standby instance.
		GRInstance Number	1 / 2

Monitoring KPIs

The following table lists monitoring KPIs.

Table 7: geo_monitoring_total KPIs

KPI Name	Description	Labels	Possible Values
geo_monitoring _total	This KPI displays the total number of successful / failure messages of different kinds such as, heartbeat / remoteNotify / TriggerGR and so on.	ControlAction Type	AdminMonitoring ActionType / AdminRemote MessageAction Type / AdminRole ChangeActionType
		ControlAction NameType	MonitorPod / MonitorBfd / RemoteMsgHeartbeat / RemoteMsgNotifyFailover / RemoteMsgNotify PrepareFailover / RemoteMsgGetSiteStatus / RemoteClusterPodFailure / RemoteSiteRole Monitoring / TriggerGRApi / ResetRoleApi
		Admin Node	Any string value. For example, GR Instance ID or instance key / pod name
		Status Code	0 / 1001 / 1002 / 1003 / 1004 / 1005 / 1006 / 1007 / 1008 / received error code (1206, 1219, 2404, …)
		Status Message	

KPI Name	Description	Labels	Possible Values
			Success (0) / STANDBY_ERROR => STANDBY/STANDBY => PRIMARY (0) / Pod Failure (0) / CLI (0) / BFD Failure (0) / Decode Failure (1001) / remote status unavailable (1002) / target role does not support (1002) / Pod Failure (1002) / CLI (1002) / BFD Failure (1002) / site is down (1003) / Pod Failure (1003) / CLI (1003) / BFD Failure (1003) / Traffic Hit (1004) / Pod Failure (1004) / CLI (1004) / BFD Failure (1004) / current role is not STANDBY_ERROR/ STANDBY to reset role (1005) / resetRole: Key not found in etcd (1006) / monitoring threshold per pod is breached (1007) / Retry on heartbeat failure (1008) / received error message (No remote host available for this request / Selected remote host <remotehostname> has no client connection / Sla is expired for transaction / …)

BFD KPIs

The following table lists BFD KPIs.

Table 8: BFD KPIs - 1

KPI Name	Description	Labels	Possible Values
bgp_speaker _bfd_status	This KPI displays BFD link status on BGP Speaker.	status	STATE_UP / STATE_DOWN
geo_bfd_ status	This KPI displays BFD link status on Geo POD.	status	STATE_UP / STATE_DOWN

Table 9: BFD KPIs - 2

KPI Name	Description	Gauge
bgp_speaker _bfd_status	This KPI displays BFD link status on BGP Speaker.	1 (UP) or 0 (DOWN)
geo_bfd_ status	This KPI displays BFD link status on Geo POD.	1 (UP) or 0 (DOWN)

Cross-rack-routing BFD Interface Monitoring

Table 10: Cross-rack-routing BFD Interface Monitoring KPIs

KPI Name	Description	Labels	Possible Values
geo_monitoring_ total	This KPI displays the total number of Gateway Down or LocalBFDInterface down messages when peer rack is down with the details of gateway IP or interface name.	ControlAction Type	AdminMonitoring ActionType
		ControlAction NameType	MonitorGateway / MonitorLocalBfdInterface
		AdminNode	gateway_ip / interface_name
		status	gateway ip is down from all proto node / local bfd interface is down from all proto node
		status_code	1012 / 1013
bgp_bfd_Monitor_ Interface_ status (Type - Gauge)	This KPI indicates each peer connection status. This connection is BFD interface configured and peers on the remote rack.	interface	<Local Rack Interface Name>
		peer_address	<Remote Rack neighbor Ip address>
		type	Bfd-Peer

KPI Name	Description	Labels	Possible Values
bgp_bfd_Monitor_ Remote_Rack_ status (Type - Gauge)	This KPI indicates the status of remote rack. Current rack interface and remote rack peers are configured in as a part of BFD peering. Rack status is up if any of the connection from both the proto node is up. If connection is down at both the proto nodes, then this KPI indicates the remote rack status is down.	status	BFD_Remote_ Rack_STATUS

Local Interface Monitoring

Table 11: Local Interface Monitoring KPI

KPI Name	Description	Labels	Possible Values
geo_monitoring_ total	This KPI displays the total number of local interface down cases with the details of interface name.	ControlAction Type	AdminMonitoring ActionType
		ControlAction NameType	MonitorInterface
		AdminNode	interface_name
		status	Local interface is down from all proto node
		status_code	1014

GR Instance Information

Table 12: GR Instance Information KPI

KPI Name	Description	Labels	Possible Values
gr_instance_ information (Type – Guage)	This KPI displays the current role of the GR instance in the application.	gr_instance_id	Configured GR instances value (numerical value)

Geo Maintenance Mode

Table 13: Geo Maintenance Mode KPI

KPI Name	Description	Labels	Possible Values
geo_MaintenanceMode_ info (Type – Guage)	This KPI displays the current state of maintenance mode for the rack.	MaintenanceMode	0: false 1: true

Bulk Statistics

The following section provides details on GR-specific bulkstats.

```
bulk-stats query GR-BGP-Incoming-Failed-Routes
 expression "sum(bgp_incoming_failedrouterequest_total) by (namespace, interface, service_IP,
 next_hop, instance_id)"
 labels     [ instance_id interface next_hop service_IP ]
 alias      gr-bgp-routes-in
exit
bulk-stats query GR-Geo-Monitoring-Failure
 expression "sum(geo_monitoring_total{ControlActionNameType=~'MonitorPod|RemoteMsgHeartbeat|
RemoteMsgGetSiteStatus|RemoteSiteRoleMonitoring|RemoteClusterPodFailure|RemoteMsgNotifyFailover|
RemoteMsgNotifyPrepareFailover|MonitorVip',status!~'success|monitoring.*'}) by (namespace,

AdminNode, ControlActionType, ControlActionNameType,  pod, status, status_code)"
 labels     [ pod AdminNode ControlActionNameType status status_code ]
 alias      gr-geo-monitoring-failure
exit
bulk-stats query GR-Geo-Monitoring-Success
 expression "sum(geo_monitoring_total{ControlActionNameType=~'MonitorPod|RemoteMsgHeartbeat|
RemoteMsgGetSiteStatus|RemoteSiteRoleMonitoring|RemoteClusterPodFailure|RemoteMsgNotifyFailover|
RemoteMsgNotifyPrepareFailover',status=~'success|monitoring.*'}) by (namespace, AdminNode,

ControlActionType, ControlActionNameType,  pod, status)"
 labels     [ pod AdminNode ControlActionNameType status ]
 alias      gr-geo-monitoring
exit
bulk-stats query GR-Geo-Monitoring-Total
 expression "sum(geo_monitoring_total{ControlActionNameType=~'MonitorPod|RemoteMsgHeartbeat|
RemoteMsgGetSiteStatus|RemoteSiteRoleMonitoring|RemoteClusterPodFailure|RemoteMsgNotifyFailover
|RemoteMsgNotifyPrepareFailover|MonitorVip'})
by (namespace, AdminNode, ControlActionType, ControlActionNameType,  pod, status)"
 labels     [ pod AdminNode ControlActionNameType status ]
 alias      gr-geo-monitoring
exit
bulk-stats query GR-Geo-Replication-Failure
 expression
"sum(geo_replication_total{ReplicationNode=~'CACHE_POD|ETCD|PEER',status!='success',
ReplicationRequestType='Response'}) by (namespace, ReplicationNode, ReplicationSyncType,
ReplicationReceiver,ReplicationRequestType,status,status_code)"
 labels     [ pod ReplicationNode ReplicationReceiver ReplicationRequestType
ReplicationSyncType status status_code ]
 alias      gr-geo-replication-failure
exit
bulk-stats query GR-Geo-Replication-Success
 expression "sum(geo_replication_total{ReplicationNode=~'CACHE_POD|ETCD|PEER',
status='success',ReplicationRequestType='Response'}) by (namespace, ReplicationNode,
ReplicationSyncType,ReplicationReceiver,ReplicationRequestType,status)"
 labels     [ pod ReplicationNode ReplicationReceiver ReplicationRequestType
```

```
ReplicationSyncType status ]
 alias      gr-geo-replication-success
exit
bulk-stats query GR-Geo-Replication-Total
 expression "sum(geo_replication_total{ReplicationNode=~'CACHE_POD|ETCD|PEER'})
by (namespace, ReplicationNode, ReplicationSyncType,ReplicationReceiver,
ReplicationRequestType,pod)"
 labels     [ pod ReplicationNode ReplicationReceiver ReplicationRequestType
ReplicationSyncType ]
 alias      gr-geo-replication-total
exit
bulk-stats query GR-Trigger-ResetRole-Api
 expression "sum(geo_monitoring_total{ControlActionNameType=~'TriggerGRApi|ResetRoleApi'})

by (namespace, AdminNode, ControlActionType, ControlActionNameType,  pod, status,
status_code)"
 labels     [ pod AdminNode ControlActionNameType status status_code ]
 alias      gr-api
exit
bulk-stats query GR-CDL-Index-Replication
 expression "sum(consumer_kafka_records_total) by (pod, origin_instance_id)"
 labels     [ origin_instance_id pod ]
 alias      gr-cdl-index-replication
exit
bulk-stats query GR-CDL-Inter-Rack-Replications-Failures
 expression "sum(datastore_requests_total{local_request='0',errorCode!='0'}) by
(operation,sliceName,errorCode)"
 labels     [ sliceName operation errorCode ]
 alias      gr-cdl-inter-rack-replications
exit
bulk-stats query GR-CDL-Inter-Rack-Replications-Success
 expression "sum(datastore_requests_total{local_request='0',errorCode='0'}) by
(operation,sliceName,errorCode)"
 labels     [ sliceName operation errorCode ]
 alias      gr-cdl-inter-rack-replications
exit
bulk-stats query GR-CDL-Inter-Rack-Replications-Total
 expression "sum(datastore_requests_total{local_request='0'}) by
(operation,sliceName,errorCode)"
 labels     [ sliceName operation errorCode ]
 alias      gr-cdl-inter-rack-replications
exit
bulk-stats query GR-CDL-Intra-Rack-Operations-Failures
 expression "sum(datastore_requests_total{local_request='1',errorCode!='0'}) by
(operation,sliceName,errorCode)"
 labels     [ sliceName operation errorCode ]
 alias      gr-cdl-intra-rack-operations
exit
bulk-stats query GR-CDL-Intra-Rack-Operations-Success
 expression "sum(datastore_requests_total{local_request='1',errorCode='0'}) by
(operation,sliceName,errorCode)"
 labels     [ sliceName operation errorCode ]
 alias      gr-cdl-intra-rack-operations
exit
bulk-stats query GR-CDL-Intra-Rack-Operations-Total
 expression "sum(datastore_requests_total{local_request='1'}) by
(operation,sliceName,errorCode)"
 labels     [ errorCode operation sliceName ]
 alias      gr-cdl-intra-rack-operations
exit
bulk-stats query GR-CDL-Session-Count-Per-Slice
 expression
sum(avg(db_records_total{namespace=~'$namespace',session_type='total'})by(systemId,sliceName))by(sliceName)
```

```
 labels     [ sliceName ]
 alias      gr-cdl-session-count-per-slice
exit
bulk-stats query GR-CDL-Session-Count-Per-System-ID
 expression sum(avg(db_records_total{namespace=~'$namespace',session_type='total'})
by(systemId,sliceName))by(systemId)
 labels     [ systemId ]
 alias      gr-cdl-session-count-per-system-id
exit
bulk-stats query GR-CDL-Slot-Records-Per-Slice
 expression "sum(slot_records_total{pod=~'.*',systemId!=''}) by (pod, sliceName)"
 labels     [ pod sliceName ]
 alias      gr-cdl-slot-records-per-slice
exit
bulk-stats query GR-CDL-Slot-Records-Per-System-ID
 expression "sum(slot_records_total{pod=~'.*',systemId!=''}) by (pod, systemId)"
 labels     [ pod systemId ]
 alias      gr-cdl-slot-records-per-system-id
exit
bulk-stats query GR-CDL-Total-Session-Count
 expression "sum(db_records_total{namespace=~'$namespace',session_type='total'}) by
(systemId,sliceName)"
 labels     [ sliceName systemId ]
 alias      gr-cdl-total-session-count
exit
```

For more information on GR-related statistics, see the following:

- In cnSGW-C statistics, you can filter GR-specific statistics using gr_instance_id label.

  For more information, see the *UCC Serving Gateway Control Plane Function - Metrics Reference*.

Alerts

The following section provides details on GR alerts.

BFD Alerts

The following table list alerts for rule group BFD with *interval-seconds* as 60.

Table 14: Alert Rule Group - BFD

Alert Rule	Severity	Duration (in mins)	Type
BFD-Link-Fail	critical	1	Communication Alarm
	Expression: sum by (namespace,pod,status) (bgp_speaker_bfd_status{status='BFD_STATUS'}) == 0 **Description:** This alert is generated when BFD link associated with BGP peering is down.		

GR Alerts

The following table list alerts for rule group GR with *interval-seconds* as 60.

Table 15: Alert Rule Group - GR

Alert Rule	Severity	Duration (in mins)	Type
Cache-POD-Replication-Immediate-Local	critical	1	Communication Alarm
	Expression: (sum by (namespace) (increase(geo_replication_total{ReplicationNode='CACHE_POD', ReplicationSyncType='Immediate',ReplicationReceiver='local', ReplicationRequestType='Response',status='success'}[1m]))/sum by (namespace) (increase(geo_replication_total{ReplicationNode='CACHE_POD', ReplicationSyncType='Immediate',ReplicationReceiver='local', ReplicationRequestType='Request'}[1m])))*100 < 90 **Description:** This alert is generated when the success rate of CACHE_POD sync type:Immediate and replication receiver:Local is below threshold value.		
Cache-POD-Replication-Immediate-Remote	critical	1	Communication Alarm
	Expression: (sum by (namespace) (increase(geo_replication_total{ReplicationNode='CACHE_POD', ReplicationSyncType='Immediate',ReplicationReceiver='remote', ReplicationRequestType='Response',status='success'}[1m]))/sum by (namespace) (increase(geo_replication_total{ReplicationNode='CACHE_POD', ReplicationSyncType='Immediate',ReplicationReceiver='remote', ReplicationRequestType='Request'}[1m])))*100 < 90 **Description:** This alert is generated when the success rate of CACHE_POD sync type:Immediate and replication receiver:Remote is below threshold value.		
Cache-POD-Replication-PULL-Remote	critical	1	Communication Alarm
	Expression: (sum by (namespace) (increase(geo_replication_total{ReplicationNode='CACHE_POD', ReplicationSyncType='PULL',ReplicationReceiver='remote', ReplicationRequestType='Response',status='success'}[1m]))/sum by (namespace) (increase(geo_replication_total{ReplicationNode='CACHE_POD', ReplicationSyncType='PULL',ReplicationReceiver='remote', ReplicationRequestType='Request'}[1m])))*100 < 90 **Description:** This alert is generated when the success rate of CACHE_POD sync type:PULL and replication receiver:Remote is below threshold value.		

Alert Rule	Severity	Duration (in mins)	Type
ETCD-Replication-Immediate-Local	critical	1	Communication Alarm
	Expression: (sum by (namespace) (increase(geo_replication_total{ReplicationNode='ETCD', ReplicationSyncType='Immediate',ReplicationReceiver='local', ReplicationRequestType='Response',status='success'}[1m]))/sum by (namespace) (increase(geo_replication_total{ReplicationNode='ETCD', ReplicationSyncType='Immediate',ReplicationReceiver='local', ReplicationRequestType='Request'}[1m])))*100 < 90 **Description:** This alert is generated when the success rate of ETCD sync type:Immediate and replication receiver:Local is below threshold value.		
ETCD-Replication-Immediate-Remote	critical	1	Communication Alarm
	Expression: (sum by (namespace) (increase(geo_replication_total{ReplicationNode='ETCD', ReplicationSyncType='Immediate',ReplicationReceiver='remote', ReplicationRequestType='Response',status='success'}[1m]))/sum by (namespace) (increase(geo_replication_total{ReplicationNode='ETCD', ReplicationSyncType='Immediate',ReplicationReceiver='remote', ReplicationRequestType='Request'}[1m])))*100 < 90 **Description:** This alert is generated when the success rate of ETCD sync type:Immediate and replication receiver:Remote is below threshold value.		
ETCD-Replication-PULL-Remote	critical	1	Communication Alarm
	Expresion: (sum by (namespace) (increase(geo_replication_total{ReplicationNode='ETCD', ReplicationSyncType='PULL',ReplicationReceiver='remote', ReplicationRequestType='Response',status='success'}[1m]))/ sum by (namespace) (increase(geo_replication_total {ReplicationNode='ETCD',ReplicationSyncType='PULL', ReplicationReceiver='remote',ReplicationRequestType= 'Request'}[1m])))*100 < 90 **Description:** This alert is generated when the success rate of ETCD sync type:PULL and replication receiver:Remote is below threshold value.		

Alert Rule	Severity	Duration (in mins)	Type
Heartbeat-Remote -Site	critical	-	Communication Alarm
	Expression: sum by (namespace) (increase(geo_monitoring_total{ControlActionNameType= 'RemoteMsgHeartbeat',status!='success'}[1m])) > 0 **Description:** This alert is triggerd when periodic Heartbeat to remote site fails.		
Local-Site-POD-Monitoring	critical	-	Communication Alarm
	Expression: sum by (namespace,AdminNode) (increase(geo_monitoring_total{ControlActionNameType ='MonitorPod'}[1m])) > 0 **Description:** This alert is triggerd when local site pod monitoring failures breaches the configured threshold for the pod mentioned in Label:{{$labels.AdminNode}}.		
PEER-Replication-Immediate-Local	critical	1	Communication Alarm
	Expression: (sum by (namespace) (increase(geo_replication_total{ReplicationNode='PEER', ReplicationSyncType='Immediate',ReplicationReceiver='local', ReplicationRequestType='Response',status='success'} [1m]))/sum by (namespace) (increase(geo_replication_total {ReplicationNode='PEER',ReplicationSyncType= 'Immediate',ReplicationReceiver='local', ReplicationRequestType='Request'}[1m])))*100 < 90 **Description:** This alert is generated when the success rate of PEER sync type:Immediate and replication receiver:Local is below threshold value.		

Alert Rule	Severity	Duration (in mins)	Type
PEER-Replication-Immediate-Remote	critical	1	Communication Alarm
	Expression: (sum by (namespace) (increase(geo_replication_total{ReplicationNode='PEER', ReplicationSyncType='Immediate',ReplicationReceiver='remote', ReplicationRequestType='Response',status='success'} [1m]))/sum by (namespace) (increase(geo_replication_total {ReplicationNode='PEER',ReplicationSyncType='Immediate', ReplicationReceiver='remote',ReplicationRequestType= 'Request'}[1m])))*100 < 90 **Description:** This alert is generated when the success rate of PEER sync type:Immediate and replication receiver:Remote is below threshold value.		
RemoteCluster-PODFailure	critical	-	Communication Alarm
	Expression: sum by (namespace,AdminNode) (increase(geo_monitoring_total{ControlActionNameType ='RemoteClusterPodFailure'}[1m])) > 0 **Description:** This alert is generated when pod failure is detected on the Remote site for the pod mentioned in Label:{{$labels.AdminNode}}.		
RemoteMsg NotifyFailover	critical	1	Communication Alarm
	Expression: sum by (namespace,status) (increase(geo_monitoring_total{ControlActionNameType ='RemoteMsgNotifyFailover',status!='success'}[1m])) > 0 **Description:** This alert is generated when transient role RemoteMsgNotifyFailover has failed for the reason mentioned in Label:{{$labels.status}}.		
RemoteMsg NotifyPrepare Failover	critical	1	Communication Alarm
	Expression: sum by (namespace,status) (increase(geo_monitoring_total{ControlActionNameType ='RemoteMsgNotifyPrepareFailover',status!='success'}[1m])) > 0 **Description:** This alert is generated when transient role RemoteMsgNotifyPrepareFailover has failed for the reason mentioned in Label:{{$labels.status}}.		

Alert Rule	Severity	Duration (in mins)	Type
RemoteSite-RoleMonitoring	critical	-	Communication Alarm
	Expression: sum by (namespace,AdminNode) (increase(geo_monitoring_total{ControlActionNameType ='RemoteSiteRoleMonitoring'}[1m])) > 0 **Description:** This alert is generated when RemoteSiteRoleMonitoring detects role inconsistency for an instance on the partner rack and accordingly changes the role of the respective instance on local rack to Primary. The impacted instance is in Label:{{$labels.AdminNode}}.		
ResetRoleApi-Initiated	critical	-	Communication Alarm
	Expression: sum by (namespace,status) (increase(geo_monitoring_total{ControlActionNameType ='ResetRoleApi'}[1m])) > 0 **Description:** This alert is generated when ResetRoleApi is initiated with the state transition of roles mentioned in Label:{{$labels.status}}.		
TriggerGRApi-Initiated	critical	-	Communication Alarm
	Expression: sum by (namespace,status) (increase(geo_monitoring_total{ControlActionNameType ='TriggerGRApi'}[1m])) > 0 **Description:** This alert is generated when TriggerGRApi is initiated for the reason mentioned in Label:{{$labels.status}}.		
VIP-Monitoring-Failures	critical	-	Communication Alarm
	Expression: sum by (namespace,AdminNode) (increase(geo_monitoring_total{ControlActionNameType ='MonitorVip'}[1m])) > 0 **Description:** This alert is generated when GR is generated upon detecting VIP monitoring failures for the VIP and Instance mentioned in the Label:{{$labels.AdminNode}}.		

CDL Alerts

The following table list alerts for rule group CDL with *interval-seconds* as 60.

Table 16: Alert Rule Group - CDL

Alert Rule	Severity	Duration (in mins)	Type
GRPC-Connections-Remote-Site	critical	1	Communication Alarm
	Expression: sum by (namespace, pod, systemId) (remote_site_connection_status) !=4 **Description:** This alert is generated when GRPC connections to remote site are not equal to 4.		
Inter-Rack -CDL-Replication	critical	1	Communication Alarm
	Expression: (sum by (namespace) (increase(datastore_requests_total{local_request=\"0\", errorCode=\"0\"}[1m]))/sum by (namespace) (increase(datastore_requests_total{local_request=\"0\"} [1m])))*100 < 90 **Description:** This alert is generated when the Inter-rack CDL replication success rate is below threshold value.		
Intra-Rack -CDL-Replication	critical	1	Communication Alarm
	Expression: (sum by (namespace) (increase(datastore_requests_total{local_request=\"1\", errorCode=\"0\"}[1m]))/sum by (namespace) (increase(datastore_requests_total{local_request=\"1\"} [1m])))*100 < 90 **Description:** This alert is generated when the Intra-rack CDL replication success rate is below threshold.		